```
> # ==========================================================================
# MAPLE TRANSCRIPT: Graviton Field Theory and Newtonian Gravity — Overlap Analysis
# Source: graviton_newtonian_overlap_EN.md
# Wave equation: (R/r)*Laplacian(H) = (1/c^2)*diff(H,t,t)
# ==========================================================================
# Structure (14 plots, one per physical section of the markdown):
#
#   SECTION 1  — Constants and derived quantities
#   SECTION 2  — Verification tables (force, Heisenberg)
#   SECTION 3  — Airy mode table
#
#   PLOT 01  [Part I-1.1]  Effective propagation speed c_eff(r)/c = sqrt(r/R)
#   PLOT 02  [Part I-1.3]  Airy eigenstates g_n(r/R), n=1,2,3  (C2=0)
#   PLOT 03  [Part I-1.4]  Quantized wave vectors and periods (mode spectrum)
#   PLOT 04  [Part II-2.2] Polynomial wave function f(r,0) and force F(r,0)
#   PLOT 05  [Part III]    Force decomposition: Newton + cosmological correction
#   PLOT 06  [Part IV-4.1] Relative deviation delta(r)=r/(2R) — linear scale
#   PLOT 07  [Part IV-4.2] Deviation delta(r) — log-log, full astronomical range
#   PLOT 08  [Part IV-4.5] Modified gravity vs MOND vs Yukawa — comparison
#   PLOT 09  [Part V]      Wave function psi(r) and probability density P(r)
#   PLOT 10  [Part V-5.3]  Graviton flux = gravitational force, 3D surface psi(r,t)
#   PLOT 11  [Part VI]     Heisenberg phase-space diagram (three graviton states)
#   PLOT 12  [Part VII]    Quantum-classical bridge: F_mod/F_Newton = 1 - r/(2R)
#   PLOT 13  [Part VIII]   Solar System precision tests — theory vs experimental limits
#   PLOT 14  [Part IX]     Constants hierarchy (log10 scale)
#
# Syntax rules applied throughout:
#   - plot() with axis[1]=[mode=log], axis[2]=[mode=log]  (NOT loglogplot)
#   - All plot ranges use dimensionless or small-integer endpoints (no large floats)
#   - Symbols: solidcircle, solidbox, diamond, cross (all valid in Maple 2024)
#   - proc() used for evaluations; quoted expressions 'f(x)' in plot()
```

```
calls
#   - seq() used instead of $ operator
#   - No Array() in plots[display] — use flat list with insequence
# ===========================================================================
======

restart;
with(plots):
with(plottools):
with(StringTools):

# ===========================================================================
======
# SECTION 1 — PHYSICAL CONSTANTS AND DERIVED QUANTITIES
# ===========================================================================
======

printf("\n%s\n SECTION 1: PHYSICAL CONSTANTS\n%s\n",
    Repeat("=",62), Repeat("=",62));

# --- Fundamental constants ---
R_v   := 3.9228e22;          # Characteristic cosmological distance [m]
c_v   := 2.99792458e8;       # Speed of light [m/s]
G_v   := 6.674e-11;          # Gravitational constant [N m^2/kg^2]
M_U   := 1.0e53;             # Estimated Universe mass [kg]
hbar  := 1.05457e-34;        # Reduced Planck constant [J s]

# --- Composite quantities ---
GM    := evalf(G_v * M_U);
c2    := evalf(2.0*Pi*c_v^2*GM / R_v^2);    # c2 = 2pi c^2 GM / R^2
N_wf  := evalf(sqrt(3.0 / (8.0*R_v^3)));    # Normalization N = sqrt(3/
(8R^3))

# --- Airy zeros ---
a1 := -2.33810741;
a2 := -4.08794944;
a3 := -5.52055983;
a4 := -6.78670809;
a5 := -7.94413359;

# --- Quantized wave vectors k_n = |a_n|^(3/2) / (2*sqrt(2)*R) ---
k1 := evalf(abs(a1)^(3/2) / (2*sqrt(2)*R_v));
k2 := evalf(abs(a2)^(3/2) / (2*sqrt(2)*R_v));
k3 := evalf(abs(a3)^(3/2) / (2*sqrt(2)*R_v));
k4 := evalf(abs(a4)^(3/2) / (2*sqrt(2)*R_v));
k5 := evalf(abs(a5)^(3/2) / (2*sqrt(2)*R_v));

# --- Oscillation quantities ---
omega1 := evalf(k1 * c_v);
```

```
T1       := evalf(2*Pi / omega1);
E1       := evalf(hbar * omega1);
m_g      := evalf(E1 / c_v^2);

# --- Heisenberg quantities ---
Delta_r   := evalf(R_v * sqrt(3.0/20.0));
Delta_p   := evalf(hbar * k1 / 2.0);
Delta_E   := evalf(hbar * omega1 / 2.0);
Delta_t   := evalf(1.0 / omega1);
hbar2     := evalf(hbar / 2.0);
UP_rm     := evalf(Delta_r * Delta_p);
UP_Et     := evalf(Delta_E * Delta_t);
sigma_r   := evalf(sqrt(hbar / (2.0*m_g*omega1)));
delta_min := evalf(hbar * c_v / E1);

printf("  R      = %12.6e  m\n",       R_v);
printf("  GM     = %12.6e  m^3/s^2\n", GM);
printf("  c2     = %12.6e  m^3/s^2\n", c2);
printf("  N_wf   = %12.6e  m^(-3/2)\n",N_wf);
printf("  k1     = %12.6e  m^-1\n",    k1);
printf("  omega1 = %12.6e  rad/s\n",   omega1);
printf("  T1     = %12.6e  s  = %.3f Myr\n", T1, evalf(T1/3.156e13));
printf("  E1     = %12.6e  J\n",        E1);
printf("  m_g   <= %12.6e  kg\n",       m_g);
printf("  Delta_r= %12.6e  m  = %.6f R\n", Delta_r, evalf(Delta_r/R_v));
printf("  Delta_p= %12.6e  kg m/s\n",   Delta_p);
printf("  sigma_r= %12.6e  m  = %.4f R\n", sigma_r, evalf(sigma_r/R_v));

# ========================================================================
======
# SECTION 2 — FORCE AND HEISENBERG VERIFICATION TABLES
# ========================================================================
======

printf("\n%s\n SECTION 2: FORCE TABLE  F_mod vs F_Newton\n%s\n",
    Repeat("=",62), Repeat("=",62));

# F_mod(r) = GM/(2R) * (r-2R)/r^2 = -GM/r^2 + GM/(2Rr)
# F_Newton(r) = -GM/r^2
# delta(r) = r/(2R)

r_test_list := [1e10, 1e15, 1e18, 1e20, 1e21, R_v, 1.5*R_v, 2.0*R_v];
r_lbl_list  := ["1e10","1e15","1e18","1e20","1e21","R","1.5R","2R"];

printf("\n  %-8s  %15s  %15s  %12s\n",
    "Scale","F_mod (N/kg)","F_Newton (N/kg)","delta=r/(2R)");
printf("  %s\n", Repeat("-",55));
for i from 1 to nops(r_test_list) do
    rv    := evalf(r_test_list[i]);
```

```
    Fmod  := evalf(GM/(2.0*R_v) * (rv - 2.0*R_v) / rv^2);
    FNewt := evalf(-GM / rv^2);
    delta := evalf(rv / (2.0*R_v));
    printf("  %-8s  %15.4e  %15.4e  %12.6f\n",
        r_lbl_list[i], Fmod, FNewt, delta);
end do;

printf("\n%s\n SECTION 2b: HEISENBERG PRODUCTS\n%s\n",
    Repeat("=",62), Repeat("=",62));
printf("  Delta_r * Delta_p = %12.6e  J s\n", UP_rm);
printf("  hbar/2            = %12.6e  J s\n", hbar2);
printf("  Ratio pos-mom     = %.6f  (linear psi; Airy exact = 1.082)\n",
    evalf(UP_rm/hbar2));
printf("  Delta_E * Delta_t = %12.6e  J s\n", UP_Et);
printf("  Ratio E-t         = %.6f  (saturated exactly = 1.0)\n",
    evalf(UP_Et/hbar2));

# ========================================================================
======
# SECTION 3 — AIRY MODE TABLE (first 5 modes)
# ========================================================================
======

printf("\n%s\n SECTION 3: AIRY MODE TABLE\n%s\n",
    Repeat("=",62), Repeat("=",62));

airy_z  := [a1,a2,a3,a4,a5];
kn_list := [k1,k2,k3,k4,k5];
En_list := [seq(evalf(hbar*kn_list[i]*c_v),             i=1..5)];
En_norm := [seq(evalf(En_list[i]/En_list[1]),           i=1..5)];
Tn_Myr  := [seq(evalf(2*Pi/(kn_list[i]*c_v)/3.156e13),  i=1..5)];

printf("\n  %-4s  %-12s  %-14s  %-12s  %-10s\n",
    "n","a_n","k_n (m^-1)","E_n/E_1","T_n (Myr)");
printf("  %s\n", Repeat("-",56));
for i from 1 to 5 do
    printf("  %-4d  %-12.6f  %-14.4e  %-12.6f  %-10.3f\n",
        i, evalf(airy_z[i]), kn_list[i], En_norm[i], Tn_Myr[i]);
end do;

# ========================================================================
====
# PLOT 01 — EFFECTIVE PROPAGATION SPEED  c_eff(r)/c = sqrt(r/R)
# [Part I, Section 1.1 — "spacetime curvature encoded in the wave
operator"]
# ========================================================================
====

printf("\n%s\n PLOT 01: Effective Propagation Speed c_eff(r)/c\n%s\n",
```

```
    Repeat("=",62), Repeat("=",62));

p01_curve := plot(sqrt(x), x=0..2,
    color="DodgerBlue", thickness=3,
    legend="c_eff(r)/c = sqrt(r/R)");

# Horizontal reference: c at r=R, c*sqrt(2) at r=2R
p01_h1 := plot(1.0, x=0..1,
    color="ForestGreen", thickness=1, linestyle=dot,
    legend="c at r=R");
p01_h2 := plot(sqrt(2.0), x=0..2,
    color="OrangeRed", thickness=1, linestyle=dashdot,
    legend="c*sqrt(2) at r=2R");

# Vertical markers at r=R and r=2R
p01_v1 := plots[display](
    plottools[line]([1.0,0.0],[1.0,1.0],
        color="ForestGreen", thickness=2, linestyle=dot));
p01_v2 := plots[display](
    plottools[line]([2.0,0.0],[2.0,sqrt(2.0)],
        color="OrangeRed", thickness=2, linestyle=dashdot));

p01_txt := plots[textplot]([
    [0.55, 1.04, "c_eff = c  at r=R"],
    [1.55, 1.46, "c_eff = c*sqrt(2)  at r=2R"],
    [0.10, 0.20, "c_eff -> 0  as r->0"]
], align={right}, font=[HELVETICA, 10]);

PLOT01 := plots[display](p01_curve, p01_h1, p01_h2, p01_v1, p01_v2,
p01_txt,
    title="PLOT 01 — Effective Graviton Propagation Speed\n"
          "c_eff(r)/c = sqrt(r/R)   [Part I, Eq. 1.1]",
    axes=boxed, gridlines=true,
    labels=["r / R", "c_eff(r) / c"],
    labeldirections=[horizontal, vertical],
    view=[0..2, 0..1.60],
    size=[800,500]);

print("PLOT 01 done."); PLOT01;

# ==========================================================================
====
# PLOT 02 — AIRY EIGENSTATES g_n(r/R), n=1,2,3
# [Part I, Sections 1.3–1.4 — g_n(x) = Ai(-|a_n|*x/2)/(x/2+eps), peak-
norm]
# ==========================================================================
====

printf("\n%s\n PLOT 02: Airy Eigenstates g_n(r/R)\n%s\n",
```

```
    Repeat("=",62), Repeat("=",62));

eps := 0.02;   # regularisation at x=0

# Sample peak amplitudes for normalisation
g1_samp := [seq(evalf(abs(AiryAi(-abs(a1)*(0.01+i*0.01)/2.0)
              /(0.01+i*0.01)/2.0+eps)), i=1..199)]:
g2_samp := [seq(evalf(abs(AiryAi(-abs(a2)*(0.01+i*0.01)/2.0)
              /(0.01+i*0.01)/2.0+eps)), i=1..199)]:
g3_samp := [seq(evalf(abs(AiryAi(-abs(a3)*(0.01+i*0.01)/2.0)
              /(0.01+i*0.01)/2.0+eps)), i=1..199)]:

g1mx := max(op(g1_samp));
g2mx := max(op(g2_samp));
g3mx := max(op(g3_samp));

p02a := plot(AiryAi(-abs(a1)*x/2.0) / (x/2.0+eps) / g1mx, x=0.05..2,
    color="DodgerBlue",  thickness=3,
    legend="n=1  a1=-2.338,  k1");
p02b := plot(AiryAi(-abs(a2)*x/2.0) / (x/2.0+eps) / g2mx, x=0.05..2,
    color="OrangeRed",   thickness=2, linestyle=dash,
    legend="n=2  a2=-4.088,  k2");
p02c := plot(AiryAi(-abs(a3)*x/2.0) / (x/2.0+eps) / g3mx, x=0.05..2,
    color="ForestGreen", thickness=2, linestyle=dashdot,
    legend="n=3  a3=-5.521,  k3");
p02d := plot(0.0, x=0..2, color="Gray", thickness=1);

# Boundary line at x=2 (g=0 condition)
p02bc := plots[display](
    plottools[line]([2.0,-1.15],[2.0,1.15],
        color="Black", thickness=2, linestyle=dot));

p02txt := plots[textplot]([
    [2.03, 0.05, "g(2R)=0"]
], align={right}, font=[HELVETICA,10]);

PLOT02 := plots[display](p02a, p02b, p02c, p02d, p02bc, p02txt,
    title="PLOT 02 — Airy Eigenstates g_n(r) of  (R/r)*Lap(g) = -k_n^2*
g\n"
          "C2=0 (regularity at origin),  g(2R)=0,  peak-normalised   [Part
I, 1.3-1.4]",
    axes=boxed, gridlines=true,
    labels=["r / R", "g_n(r)  [peak-normalised]"],
    labeldirections=[horizontal, vertical],
    view=[0..2.1, -1.2..1.2],
    size=[800,500]);

print("PLOT 02 done."); PLOT02;
```

```
# ======================================================================
====
# PLOT 03 — MODE SPECTRUM: normalised energies E_n/E_1 for n=1..5
# [Part I, Section 1.4 — quantization table]
# ======================================================================
====

printf("\n%s\n PLOT 03: Mode Spectrum E_n/E_1 (first 5 Airy modes)\n%s\n",
    Repeat("=",62), Repeat("=",62));

# Horizontal bar chart of E_n/E_1
p03pts := plots[listplot](En_norm,
    style=point, symbol=solidbox, symbolsize=16,
    color="DodgerBlue",
    legend="E_n / E_1");

p03bars := [seq(
    plottools[line]([i-0.35, En_norm[i]], [i+0.35, En_norm[i]],
        color="OrangeRed", thickness=4),
    i=1..5)]:

# Period labels above each bar
p03txt := plots[textplot](
    [seq([i, En_norm[i]+0.07,
          cat("T=", convert(round(evalf(Tn_Myr[i])),string), " Myr")],
     i=1..5)],
    align={above}, font=[HELVETICA, 9]);

# Vertical lines from axis to each point
p03vlines := plots[display](seq(
    plottools[line]([i, 0.0],[i, En_norm[i]],
        color="LightGray", thickness=1, linestyle=dot),
    i=1..5)):

PLOT03 := plots[display](p03pts, op(p03bars), p03vlines, p03txt,
    title="PLOT 03 — Graviton Energy Spectrum: First 5 Airy Modes\n"
          "E_n = hbar*|a_n|^(3/2)/(2*sqrt(2)*R)*c,  quantized by Airy
zeros   [Part I, 1.4]",
    axes=boxed, gridlines=true,
    labels=["Mode n", "E_n / E_1"],
    labeldirections=[horizontal, vertical],
    view=[0..6, 0..max(op(En_norm))+0.4],
    size=[800,500]);

print("PLOT 03 done."); PLOT03;

# ======================================================================
====
# PLOT 04 — POLYNOMIAL WAVE FUNCTION f(r,0) AND GRAVITON FLUX F(r,0)
```

```
# [Part II, Sections 2.1–2.2]
# f(x) = sqrt(3/8)*(x-2), F(x) = (x-2)/(2x^2), x=r/R dimensionless
# ========================================================================
====

printf("\n%s\n PLOT 04: Polynomial Wave Function and Graviton Flux\n%s\n",
    Repeat("=",62), Repeat("=",62));

# Normalised wave function: psi(x) = sqrt(3/8)*(x-2)
p04psi := plot(sqrt(3.0/8.0)*(x-2.0), x=0..2,
    color="DodgerBlue", thickness=3,
    legend="psi(r/R) = N*(r-2R)  [left axis]");

# Flux / force scaled by 2 for visibility: F(x)*2*x^2 = (x-2)
p04F := plot((x-2.0)/(2.0*x^2+0.001), x=0.05..1.95,
    color="OrangeRed", thickness=2, linestyle=dash,
    legend="F(r)*R^2/GM = (x-2)/(2x^2)  [right axis scaled]");

# Mark zero at x=2: the boundary condition f(2R)=0
p04bc := plots[display](
    plottools[line]([2.0,-0.75],[2.0,0.15],
        color="Black", thickness=2, linestyle=dot));

p04z := plot(0.0, x=0..2, color="Gray", thickness=1);

p04txt := plots[textplot]([
    [2.04, 0.08, "f(2R)=0"],
    [0.20, -0.55,"f(r,0)=c2*(r-2R)"]
], align={right}, font=[HELVETICA,9]);

PLOT04 := plots[display](p04psi, p04F, p04bc, p04z, p04txt,
    title="PLOT 04 — Polynomial Wave Function f(r,0) and Graviton Flux F
(r,0)\n"
          "f(x)=N*(x-2),  F*R^2/GM=(x-2)/(2x^2),  x=r/R   [Part II, 2.1
-2.2]",
    axes=boxed, gridlines=true,
    labels=["r / R", "f(r,0) normalised  |  F(r)*R^2/GM"],
    labeldirections=[horizontal, vertical],
    view=[0..2, -0.80..0.50],
    size=[800,500]);

print("PLOT 04 done."); PLOT04;

# ========================================================================
====
# PLOT 05 — FORCE DECOMPOSITION: Newton + cosmological correction
# [Part III, Section 3.4]
# All in units of GM/R^2, signed, x=r/R, linear scale
# F_mod(x)  = (x-2)/(2x^2)
```

```
# F_Newton   = -1/x^2
# F_cosmo    = +1/(2x)    [repulsive correction]
# ==========================================================================
====

printf("\n%s\n PLOT 05: Force Decomposition F_mod = F_Newton +
F_cosmo\n%s\n",
    Repeat("=",62), Repeat("=",62));

p05mod := plot((x-2.0)/(2.0*x^2),  x=0.06..1.95,
    color="DodgerBlue",  thickness=3,
    legend="F_modified = F_Newton + F_cosmo");
p05new := plot(-1.0/x^2,            x=0.06..1.95,
    color="OrangeRed",   thickness=2, linestyle=dash,
    legend="F_Newton = -GM/r^2");
p05cos := plot(1.0/(2.0*x),         x=0.06..1.95,
    color="ForestGreen", thickness=2, linestyle=dashdot,
    legend="F_cosmo = +GM/(2Rr)   [repulsion]");
p05zer := plot(0.0, x=0..2, color="Gray", thickness=1);

# Mark r=R where F_mod = F_Newton/2
p05vR := plots[display](
    plottools[line]([1.0,-22.0],[1.0,6.0],
        color="Purple", thickness=1, linestyle=dot));
p05txt := plots[textplot]([
    [1.04, 5.0, "r=R:  F_mod = F_N/2"],
    [1.04,-20.5,"F_mod(2R) = 0"]
], align={right}, font=[HELVETICA,9]);

PLOT05 := plots[display](p05mod, p05new, p05cos, p05zer, p05vR, p05txt,
    title="PLOT 05 — Force Decomposition  F_mod = F_Newton +
F_cosmological\n"
          "Units GM/R^2,  x=r/R,  exact: F_mod(2R)=0   [Part III, 3.4]",
    axes=boxed, gridlines=true,
    labels=["r / R", "Force * R^2 / GM"],
    labeldirections=[horizontal, vertical],
    view=[0..2, -24..8],
    size=[800,500]);

print("PLOT 05 done."); PLOT05;

# ==========================================================================
====
# PLOT 06 — RELATIVE DEVIATION delta(r) = r/(2R), LINEAR SCALE
# [Part IV, Section 4.1 — "the single most important result"]
# Shows the key regimes: Solar System, Galactic, Cosmological
# ==========================================================================
====
```

```
printf("\n%s\n PLOT 06: Relative Deviation delta(r)=r/(2R)  [linear]
\n%s\n",
    Repeat("=",62), Repeat("=",62));

p06line := plot(x/2.0, x=0..2,
    color="DodgerBlue", thickness=3,
    legend="delta(r) = r/(2R)  [exact, no free parameters]");
p06ref  := plot(1.0, x=0..2,
    color="Black", thickness=1, linestyle=solid);
p06half := plot(0.5, x=0..2,
    color="OrangeRed", thickness=1, linestyle=dash,
    legend="50% at r=R");
p06gal  := plot(0.00127, x=0..2,
    color="ForestGreen", thickness=1, linestyle=dashdot,
    legend="0.127% at r=10^20 m  (galactic)");

# Mark r=R and r=2R
p06vR := plots[display](
    plottools[line]([1.0,0.0],[1.0,0.5],
        color="OrangeRed", thickness=2, linestyle=dot),
    plottools[line]([0.0,0.5],[1.0,0.5],
        color="OrangeRed", thickness=2, linestyle=dot));
p06v2R := plots[display](
    plottools[line]([2.0,0.0],[2.0,1.0],
        color="DodgerBlue", thickness=2, linestyle=dot));

p06txt := plots[textplot]([
    [1.04, 0.52, "delta=0.50  at r=R"],
    [2.04, 1.02, "delta=1.00  at r=2R"],
    [0.10, 0.04, "delta<<1  Solar System"]
], align={right}, font=[HELVETICA,9]);

PLOT06 := plots[display](p06line, p06ref, p06half, p06gal,
    p06vR, p06v2R, p06txt,
    title="PLOT 06 — Relative Deviation from Newton: delta(r) = r/(2R)\n"
          "Exact result, no free parameters   [Part IV, 4.1]",
    axes=boxed, gridlines=true,
    labels=["r / R", "delta(r) = [F_mod - F_N] / |F_N|"],
    labeldirections=[horizontal, vertical],
    view=[0..2.1, 0..1.08],
    size=[800,500]);

print("PLOT 06 done."); PLOT06;

# ============================================================================
====
# PLOT 07 — DEVIATION delta(r) LOG-LOG, FULL ASTRONOMICAL RANGE
# [Part IV, Section 4.2 — regime analysis]
# x=r/R from 1e-14 to 2; horizontal lines = experimental precision limits
```

```
# ==============================================================
====

printf("\n%s\n PLOT 07: delta(r) log-log full astronomical range\n%s\n",
    Repeat("=",62), Repeat("=",62));

p07main := plot(x/2.0, x=1e-14..1.999,
    color="DodgerBlue", thickness=3,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="delta(r) = r/(2R)");

# Experimental precision limits (horizontal dashed lines)
p07lim1 := plot(1e-5,  x=1e-14..1.999,
    color="OrangeRed", thickness=1, linestyle=dashdot,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="Solar System limit ~ 1e-5");
p07lim2 := plot(1e-11, x=1e-14..1.999,
    color="ForestGreen", thickness=1, linestyle=dash,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="Lunar Ranging limit ~ 1e-11");
p07lim3 := plot(1e-14, x=1e-14..1.999,
    color="Purple", thickness=1, linestyle=dot,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="Binary Pulsar limit ~ 1e-14");

# Key scale markers (vertical gray lines)
r_keys := [3.84e8/R_v, 5.79e10/R_v, 1e20/R_v, 1.0]:
p07vlines := plots[display](seq(
    plottools[line]([r_keys[i],1e-16],[r_keys[i],2.0],
        color="LightGray", thickness=1),
    i=1..4)):

PLOT07 := plots[display](p07main, p07lim1, p07lim2, p07lim3, p07vlines,
    title="PLOT 07 — Deviation delta(r)=r/(2R) vs Scale  (log-log)\n"
          "Horizontal lines: experimental precision limits   [Part IV,
4.2]",
    axes=boxed, gridlines=true,
    labels=["r/R  [log]", "delta(r)  [log]"],
    labeldirections=[horizontal, vertical],
    size=[800,500]);

print("PLOT 07 done."); PLOT07;

# ==============================================================
====
# PLOT 08 — MODIFIED GRAVITY vs MOND vs YUKAWA COMPARISON
# [Part IV, Section 4.5]
# All forces normalised to |F_Newton(x)| = 1, x=r/R, log-log
# This work: F_mod/|F_N| = 1 - x/2
```

```
# MOND:        F_MOND/|F_N| ~ sqrt(x^2 * R_v^2 / r_M^2) for large x
[approx]
# Yukawa:      F_Y/|F_N|    = exp(-lambda*x)*(1 + lambda*x)  with lambda=0.5
# ==========================================================================
====

printf("\n%s\n PLOT 08: Modified Gravity vs MOND vs Yukawa\n%s\n",
    Repeat("=",62), Repeat("=",62));

# This work: ratio = 1 - x/2  (= F_mod/|F_Newton|)
p08tw := plot(1.0 - x/2.0, x=0.001..1.999,
    color="DodgerBlue", thickness=3,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="This work: (1 - r/2R)");

# Newton reference: ratio = 1
p08new := plot(1.0, x=0.001..2.0,
    color="Black", thickness=2, linestyle=dash,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="Newton: ratio = 1");

# Yukawa: F_Y = -GM*exp(-r/lambda)/r^2*(1+r/lambda)
# In ratio to Newton: exp(-x*lam)*(1+x*lam), lam=0.5 (lambda=0.5*R)
lam_Y := 0.5:
p08yuk := plot(exp(-x*lam_Y)*(1.0+x*lam_Y), x=0.001..1.999,
    color="ForestGreen", thickness=2, linestyle=dashdot,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="Yukawa (lambda=0.5R)");

# MOND (deep MOND regime interpolation): |F_MOND/F_N| ~ sqrt(|F_N|/a0) for
F<<a0
# With a0 in dimensionless units a0_dim = a0*R_v^2/GM = 1e-10*R_v^2/GM
a0_dim := evalf(1e-10 * R_v^2 / GM):  # ~2.3e-2
p08mond := plot(sqrt(a0_dim/(x^2)), x=0.001..1.999,
    color="OrangeRed", thickness=2, linestyle=dot,
    axis[1]=[mode=log], axis[2]=[mode=log],
    legend="MOND deep regime: sqrt(a0*R^2/(GM*x^2))");

PLOT08 := plots[display](p08tw, p08new, p08yuk, p08mond,
    title="PLOT 08 — Modified Gravity Comparison: This Work vs MOND vs
Yukawa\n"
          "Ratio |F_theory|/|F_Newton|,  x=r/R,  log-log   [Part IV, 4.5]
",
    axes=boxed, gridlines=true,
    labels=["r/R  [log]", "|F_theory| / |F_Newton|  [log]"],
    labeldirections=[horizontal, vertical],
    size=[800,500]);

print("PLOT 08 done."); PLOT08;
```

```
# ==================================================================
====
# PLOT 09 — WAVE FUNCTION psi(r) AND PROBABILITY DENSITY P(r)
# [Part V, Sections 5.1–5.2 — normalised on r in [0,2R]]
# Shows: psi(x)=sqrt(3/8)*(x-2), P(x)=(3/8)*(x-2)^2, <r>, Delta_r markers
# ==================================================================
====

printf("\n%s\n PLOT 09: Wave Function psi(r) and Probability
Density\n%s\n",
    Repeat("=",62), Repeat("=",62));

p09psi := plot(sqrt(3.0/8.0)*(x-2.0), x=0..2,
    color="DodgerBlue", thickness=3,
    legend="psi(r/R) = N*(r-2R)");
p09P   := plot((3.0/8.0)*(x-2.0)^2,  x=0..2,
    color="OrangeRed",   thickness=3, linestyle=dash,
    legend="|psi|^2 = P(r/R) = N^2*(r-2R)^2");
p09zer := plot(0.0, x=0..2, color="Gray", thickness=1);

# <r>/R = 0.5 marker
p09mr := plots[display](
    plottools[line]([0.5,-0.65],[0.5,0.85],
        color="ForestGreen", thickness=2, linestyle=dot));

# Delta_r boundaries: x = 0.5 +/- sqrt(3/20)
Dr_sc := evalf(sqrt(3.0/20.0));   # ~0.3873
p09D  := plots[display](
    plottools[line]([0.5-Dr_sc, 0.0],[0.5-Dr_sc, 0.45],
        color="Purple", thickness=2, linestyle=dashdot),
    plottools[line]([0.5+Dr_sc, 0.0],[0.5+Dr_sc, 0.45],
        color="Purple", thickness=2, linestyle=dashdot));

p09txt := plots[textplot]([
    [0.52, 0.78, "<r> = R/2"],
    [0.5-Dr_sc-0.04, 0.48, "<r>-Dr"],
    [0.5+Dr_sc+0.02, 0.48, "<r>+Dr"]
], align={right}, font=[HELVETICA,9]);

PLOT09 := plots[display](p09psi, p09P, p09zer, p09mr, p09D, p09txt,
    title="PLOT 09 — Graviton Wave Function psi(r) and Probability
Density P(r)\n"
          "<r>=R/2,  Delta_r=R*sqrt(3/20)=0.387R   [Part V, 5.1-5.2]",
    axes=boxed, gridlines=true,
    labels=["r / R", "psi(r/R)  or  P(r/R) = |psi|^2"],
    labeldirections=[horizontal, vertical],
    view=[0..2, -0.70..0.90],
    size=[800,500]);
```

```
print("PLOT 09 done."); PLOT09;

# ========================================================================
====
# PLOT 10 — SPACETIME EVOLUTION psi(r,t)  [3D surface]
# [Part V, Section 5.3 — "graviton flux = gravitational force"]
# psi(x,tau) = sqrt(3/8)*(x-2)*cos(2*Pi*tau), x=r/R, tau=t/T1
# ========================================================================
====

printf("\n%s\n PLOT 10: Spacetime Evolution psi(r,t) — 3D surface\n%s\n",
    Repeat("=",62), Repeat("=",62));

PLOT10 := plot3d(sqrt(3.0/8.0)*(x-2.0)*cos(2.0*Pi*tau),
    x=0..2, tau=0..1,
    grid=[45,35], style=surface, shading=zhue,
    axes=boxed, orientation=[40,65],
    title="PLOT 10 — Graviton Field psi(r,t) = N*(r-2R)*cos(omega1*t)
[3D]\n"
          "x=r/R in [0,2],  tau=t/T_1 in [0,1]   [Part V, 5.3]",
    labels=["r/R", "t/T_1", "psi(r,t)"],
    labeldirections=[horizontal, horizontal, vertical],
    colorscheme=["zgradient",["DodgerBlue","White","OrangeRed"]],
    size=[800,600]);

print("PLOT 10 done."); PLOT10;

# ========================================================================
====
# PLOT 11 — HEISENBERG PHASE-SPACE DIAGRAM (three graviton states)
# [Part VI, Section 6.3]
# Axes: (Delta_r normalised, product/hbar2)
# Three points: linear-psi (0.490), Airy exact (1.082), coherent (1.000)
# ========================================================================
====

printf("\n%s\n PLOT 11: Heisenberg Phase-Space Diagram\n%s\n",
    Repeat("=",62), Repeat("=",62));

ratio_lin  := evalf(UP_rm / hbar2);    # ~0.490
ratio_Airy := 1.082;
sigma_norm := evalf(sigma_r / Delta_r);

# Heisenberg hyperbola: y(x) = ratio_lin / x
p11hyp := plot(ratio_lin / x, x=0.05..8.0,
    color="Gray", thickness=2, linestyle=dash,
    legend="Heisenberg bound  Dr*Dp = hbar/2");

# Minimum bound line
```

```
p11min := plot(1.0, x=0.05..8.0,
    color="Black", thickness=1, linestyle=solid,
    legend="Minimum (ratio=1)");

# Linear-psi state
p11lin := plots[pointplot]([[1.0, ratio_lin]],
    symbol=solidcircle, symbolsize=20,
    color="DodgerBlue",
    legend=cat("Linear psi  (ratio=",
               convert(evalf(ratio_lin,4),string),")"));

# Exact Airy state
p11air := plots[pointplot]([[1.07, ratio_Airy]],
    symbol=solidbox, symbolsize=20,
    color="OrangeRed",
    legend="Exact Airy state  (ratio=1.082)");

# Coherent state (minimum uncertainty)
p11coh := plots[pointplot]([[sigma_norm, 1.0]],
    symbol=diamond, symbolsize=24,
    color="ForestGreen",
    legend="Coherent state  (ratio=1.000, minimum)");

p11txt := plots[textplot]([
    [1.18, ratio_lin+0.07, "Linear psi"],
    [1.24, ratio_Airy+0.08,"Airy exact"],
    [sigma_norm+0.3, 1.08,  "Coherent min."]
], align={right}, font=[HELVETICA,9]);

PLOT11 := plots[display](p11hyp, p11min, p11lin, p11air, p11coh, p11txt,
    title="PLOT 11 — Heisenberg Phase-Space: (Delta_r*Delta_p)/(hbar/2)
\n"
          "Three graviton states; E-t uncertainty saturated exactly
[Part VI, 6.3]",
    axes=boxed, gridlines=true,
    labels=["Delta_r scale factor", "(Delta_r * Delta_p) / (hbar/2)"],
    labeldirections=[horizontal, vertical],
    view=[0..8, 0..2.0],
    size=[800,500]);

print("PLOT 11 done."); PLOT11;

# ======================================================================
====
# PLOT 12 — QUANTUM-CLASSICAL BRIDGE: F_mod/|F_Newton| = 1 - r/(2R)
# [Part VII, Sections 7.1–7.2]
# The classical limit is recovered exactly for r<<2R
# ======================================================================
====
```

```
printf("\n%s\n PLOT 12: Quantum-Classical Bridge\n%s\n",
    Repeat("=",62), Repeat("=",62));

p12rat := plot(1.0 - x/2.0, x=0..2,
    color="DodgerBlue", thickness=3,
    legend="F_mod / |F_Newton| = 1 - r/(2R)");
p12new := plot(1.0, x=0..2,
    color="OrangeRed", thickness=2, linestyle=dash,
    legend="Newton  (ratio = 1)");
p12zer := plot(0.0, x=0..2,
    color="Black", thickness=1);

# Fill region between 0 and curve — show deviation area
# Shade via polygon approximation
shade_pts := [seq([i*2.0/200, 0.0], i=0..200),
              seq([(200-i)*2.0/200, 1.0-(200-i)*2.0/200/2.0], i=0..200)]:
p12shade := plots[polygonplot](shade_pts,
    color="LightBlue", transparency=0.6,
    style=patchnogrid):

# Vertical marker at r=R (50% line)
p12vR := plots[display](
    plottools[line]([1.0,0.0],[1.0,0.5],
        color="ForestGreen", thickness=2, linestyle=dot),
    plottools[line]([0.0,0.5],[1.0,0.5],
        color="ForestGreen", thickness=2, linestyle=dot));

p12txt := plots[textplot]([
    [1.05, 0.53, "50% reduction at r=R"],
    [0.10, 0.96, "~100% Newton  (r<<R)"],
    [1.70, 0.20, "Cosmological regime"]
], align={right}, font=[HELVETICA,9]);

PLOT12 := plots[display](p12shade, p12rat, p12new, p12zer, p12vR, p12txt,
    title="PLOT 12 — Quantum-Classical Bridge: F_mod/|F_Newton| = 1 - r/
(2R)\n"
          "Shaded region = deviation from Newton   [Part VII, 7.1-7.2]",
    axes=boxed, gridlines=true,
    labels=["r / R", "F_modified / |F_Newton|"],
    labeldirections=[horizontal, vertical],
    view=[0..2, 0..1.08],
    size=[800,500]);

print("PLOT 12 done."); PLOT12;

# ==========================================================================
====
# PLOT 13 — SOLAR SYSTEM PRECISION TESTS: theory delta vs experimental
limits
```

```
# [Part VIII, Section 8.4]
# Uses solidcircle for theory, solidbox for limit; connected by dotted
line
# ==========================================================================
====

printf("\n%s\n PLOT 13: Solar System Precision Tests\n%s\n",
    Repeat("=",62), Repeat("=",62));

# Test data from Part VIII Table (log10 values)
test_idx  := [1, 2, 3, 4, 5, 6, 7]:
test_name := ["Lunar LR","Bin.Pulsar","Solar(in)","Mercury",
               "Galactic","Clusters","r=R  (pred.)"]:
delta_th  := [4.9e-15, 1.3e-14, 1.27e-13, 7.4e-13, 1.27e-3, 0.13, 0.50]:
limit_ex  := [1e-11, 1e-14, 1e-5, 1e-5, 1e-2, 0.30, -1]:  # -1=no limit

d_log := [seq(evalf(log10(delta_th[i])),i=1..7)]:
l_log := [seq(`if`(limit_ex[i]>0, evalf(log10(limit_ex[i])), 999),
              i=1..7)]:

# Theory points
p13th := plots[pointplot](
    [seq([test_idx[i], d_log[i]], i=1..7)],
    symbol=solidcircle, symbolsize=18, color="DodgerBlue",
    legend="log10(delta_theory)");

# Experimental limits (only 6 have limits)
p13lm := plots[pointplot](
    [seq([test_idx[i], l_log[i]], i=1..6)],
    symbol=solidbox, symbolsize=18, color="OrangeRed",
    legend="log10(experimental limit)");

# Connecting lines: theory to limit (dashed, shows margin)
p13conn := plots[display](seq(
    `if`(limit_ex[i]>0,
        plottools[line]([i, d_log[i]], [i, l_log[i]],
            color="Gray", thickness=1, linestyle=dot),
        plot(0,x=0..0.01,color="White")),
    i=1..6)):

# Labels below each theory point
p13txt := plots[textplot](
    [seq([test_idx[i], d_log[i]-0.8, test_name[i]], i=1..7)],
    align={below}, font=[HELVETICA,8]);

# Safety zone: green fill between curves for tests 1,2,3,4 (safe margin)
p13zer := plot(-15.5, x=0..8, color="Gray", thickness=1, linestyle=dot):

PLOT13 := plots[display](p13th, p13lm, p13conn, p13txt, p13zer,
```

```
    title="PLOT 13 — Solar System and Astronomical Precision Tests\n"
          "Blue=theory delta(r),  Red=experimental limit  [log10 scale]
[Part VIII]",
    axes=boxed, gridlines=true,
    labels=["Test index", "log10(delta)"],
    labeldirections=[horizontal, vertical],
    view=[0..8, -16..1],
    size=[800,500]);

print("PLOT 13 done."); PLOT13;

# =========================================================================
======
# FULL SUMMARY TABLE
# =========================================================================
======

printf("\n%s\n FULL SUMMARY TABLE\n%s\n",
    Repeat("=",62), Repeat("=",62));

printf("%-44s  %20s  %12s\n","Quantity","Value","Unit");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20.6e  %12s\n","R (cosmological distance)",    R_v,
           "m");
printf("%-44s  %20.6e  %12s\n","c (speed of light)",           c_v,
           "m/s");
printf("%-44s  %20.6e  %12s\n","G (Newton constant)",          G_v,
           "N m^2/kg^2");
printf("%-44s  %20.6e  %12s\n","hbar",                         hbar,
           "J s");
printf("%-44s  %20.6e  %12s\n","GM",                           GM,
           "m^3/s^2");
printf("%-44s  %20.6e  %12s\n","c2 = 2pi c^2 GM/R^2",         c2,
          "m^3/s^2");
printf("%-44s  %20s  %12s\n",  "c1",                           "0  (exact)
",           "m^2/s^2");
printf("%-44s  %20.6e  %12s\n","N (normalization)",            N_wf,
           "m^(-3/2)");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20.6e  %12s\n","k1 (fund. wave vector)",       k1,
           "m^-1");
printf("%-44s  %20.6e  %12s\n","k2",                           k2,
           "m^-1");
printf("%-44s  %20.6e  %12s\n","k3",                           k3,
           "m^-1");
printf("%-44s  %20.6e  %12s\n","omega1",                       omega1,
           "rad/s");
printf("%-44s  %20.6e  %12s\n","T1 (fund. period)",            T1,
           "s");
```

```
printf("%-44s  %20.6f  %12s\n","T1",                       evalf
(T1/3.156e13),    "Myr");
printf("%-44s  %20.6e  %12s\n","E1 (fund. graviton energy)",   E1,
          "J");
printf("%-44s  %20.6e  %12s\n","m_g (graviton mass bound)",    m_g,
          "kg");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20.6e  %12s\n","<r> = R/2",                evalf
(R_v/2),          "m");
printf("%-44s  %20.6e  %12s\n","Delta_r = R*sqrt(3/20)",       Delta_r,
          "m");
printf("%-44s  %20.6f  %12s\n","Delta_r / R",              evalf
(Delta_r/R_v),    "");
printf("%-44s  %20.6e  %12s\n","Delta_p = hbar*k1/2",          Delta_p,
          "kg m/s");
printf("%-44s  %20.6e  %12s\n","Delta_r * Delta_p",            UP_rm,
          "J s");
printf("%-44s  %20.6e  %12s\n","hbar/2",                       hbar2,
          "J s");
printf("%-44s  %20.6f  %12s\n","Ratio (pos-mom)/(hbar/2)",     evalf
(UP_rm/hbar2),    "");
printf("%-44s  %20.6f  %12s\n","Ratio Airy exact",             1.082,
          "");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20.6e  %12s\n","Delta_E = hbar*omega1/2",      Delta_E,
          "J");
printf("%-44s  %20.6e  %12s\n","Delta_t = 1/omega1",           Delta_t,
          "s");
printf("%-44s  %20.6f  %12s\n","Delta_t",                      evalf
(Delta_t/3.156e13),"Myr");
printf("%-44s  %20.6e  %12s\n","Delta_E * Delta_t",            UP_Et,
          "J s");
printf("%-44s  %20.6f  %12s\n","Ratio (E-t)/(hbar/2)",         evalf
(UP_Et/hbar2),    "");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20.6e  %12s\n","sigma_r (coherent state)",     sigma_r,
          "m");
printf("%-44s  %20.6f  %12s\n","sigma_r / R",                  evalf
(sigma_r/R_v),    "");
printf("%-44s  %20.6e  %12s\n","delta_r_min = hbar*c/E1",      delta_min,
          "m");
printf("%-44s  %20.6f  %12s\n","delta_r_min / R",              evalf
(delta_min/R_v),  "");
printf("%s\n", Repeat("-",80));
printf("%-44s  %20s  %12s\n","delta(r)",                       "r/(2R)
exact","");
printf("%-44s  %20.6f  %12s\n","delta(r=R)",                   0.5,
          "");
printf("%-44s  %20.6f  %12s\n","F_mod(R) / F_Newton(R)",       0.5,
```

```
           "");
printf("%-44s  %20.6f  %12s\n","F_mod(2R)  [N/kg]",           0.0,
           "N/kg");
printf("%s\n", Repeat("-",80));

# ===========================================================================
====
# PLOT 14 — PHYSICAL CONSTANTS HIERARCHY (log10 scale)
# [Part IX — reference table visualised]
# All variable names prefixed p14_ to avoid any collision with the summary
# ===========================================================================
====

printf("\n%s\n PLOT 14: Physical Constants Hierarchy (log10 scale)\n%s\n",
    Repeat("=",62), Repeat("=",62));

p14_names := [
    "R [m]","c [m/s]","GM","c2",
    "k1 [1/m]","omega1","E1 [J]","m_g [kg]",
    "Delta_r","Delta_p","hbar","sigma_r"]:

p14_vals := [
    evalf(log10(R_v)),
    evalf(log10(c_v)),
    evalf(log10(GM)),
    evalf(log10(c2)),
    evalf(log10(k1)),
    evalf(log10(omega1)),
    evalf(log10(E1)),
    evalf(log10(m_g)),
    evalf(log10(Delta_r)),
    evalf(log10(Delta_p)),
    evalf(log10(hbar)),
    evalf(log10(sigma_r))]:

p14_n := 12:

p14_colors := ["DodgerBlue","DodgerBlue","OrangeRed","OrangeRed",
               "ForestGreen","ForestGreen","Purple","Purple",
               "Goldenrod","Goldenrod","DarkRed","DarkCyan"]:

# Points — no legend= here; labels are drawn as inline textplot below
p14pts := plots[display](seq(
    plots[pointplot]([[i, p14_vals[i]]],
        symbol=solidbox, symbolsize=18,
        color=p14_colors[i]),
    i=1..p14_n)):

p14zer := plot(0.0, x=0..p14_n+1,
```

```
    color="Black", thickness=1, linestyle=dash):

p14stems := plots[display](seq(
    plottools[line]([i, 0.0],[i, p14_vals[i]],
        color="LightGray", thickness=1),
    i=1..p14_n)):

# Inline labels: above positive values, below negative — staggered
vertically
# to prevent overlap: even indices get extra +4 / -5 offset
p14txt := plots[textplot](
    [seq([i,
          p14_vals[i] +
              `if`(p14_vals[i] >= 0,
                   `if`(irem(i,2)=0, 6.0, 2.5),
                   `if`(irem(i,2)=0, -6.0, -3.0)),
          p14_names[i]],
     i=1..p14_n)],
    align={above}, font=[HELVETICA, 8]):

PLOT14 := plots[display](p14stems, p14pts, p14zer, p14txt,
    title="PLOT 14 — Physical Constants Hierarchy: log10(value in SI)\n"
          "Key quantities of the graviton-Newton overlap framework   [Part
IX]",
    axes=boxed, gridlines=false,
    labels=["Quantity index", "log10(value in SI units)"],
    labeldirections=[horizontal, vertical],
    view=[0..p14_n+1, -75..58],
    size=[1100,560]);

print("PLOT 14 done."); PLOT14;

# ==========================================================================
======
# COMBINED DISPLAY — 2D plots in slideshow (PLOT14 included)
# Separated from PLOT14 by a clear header
# ==========================================================================
======

printf("\n%s\n%s\n COMBINED DISPLAY (2D — insequence)\n"
       " Use arrow keys in Maple to cycle through all 13 two-dimensional
plots.\n"
       "%s\n",
    Repeat("=",62), Repeat(" ",62), Repeat("=",62));

COMBINED_2D := plots[display](
    [PLOT01, PLOT02, PLOT03, PLOT04, PLOT05, PLOT06,
     PLOT07, PLOT08, PLOT09, PLOT11, PLOT12, PLOT13, PLOT14],
    insequence=true,
```

```
    title="Graviton-Newtonian Overlap: All 2D Plots (use arrow keys)");

print("Combined 2D display ready (13 slides).");
COMBINED_2D;

# ==========================================================================
======
# PLOT 10 — 3D surface  (displayed separately, after a clear separator)
# ==========================================================================
======

printf("\n%s\n PLOT 10 (3D) — Graviton Field psi(r,t) — displayed
separately\n%s\n",
    Repeat("=",62), Repeat("=",62));

print("3D spacetime surface:"); PLOT10;

printf("\n%s\n TRANSCRIPT COMPLETE\n"
       " Plots: PLOT01..PLOT14  (13 two-dimensional + 1 three-dimensional)
\n"
       " Source: graviton_newtonian_overlap_EN.md\n%s\n",
    Repeat("=",62), Repeat("=",62));
```

```
==============================================================
 SECTION 1: PHYSICAL CONSTANTS
==============================================================
```

$$R\_v := 3.9228 \times 10^{22}$$

$$c\_v := 2.99792458 \times 10^{8}$$

$$G\_v := 6.674 \times 10^{-11}$$

$$M\_U := 1.0 \times 10^{53}$$

$$\hbar := 1.05457 \times 10^{-34}$$

$$GM := 6.6740 \times 10^{42}$$

$$c2 := 2.449148640 \times 10^{15}$$

$$N\_wf := 7.881726924 \times 10^{-35}$$

$$a1 := -2.33810741$$

$$a2 := -4.08794944$$

$$a3 := -5.52055983$$

$$a4 := -6.78670809$$

$$a5 := -7.94413359$$

$$k1 := 3.222222512 \times 10^{-23}$$

$$k2 := 7.449328282 \times 10^{-23}$$

$$k3 := 1.169051172 \times 10^{-22}$$

$$k4 := 1.593483586 \times 10^{-22}$$

$$k5 := 2.018034783 \times 10^{-22}$$

$$\omega 1 := 9.659980071 \times 10^{-15}$$

$$T1 := 6.504346036 \times 10^{14}$$

$$E1 := 1.018712518 \times 10^{-48}$$

$$m\_g := 1.133470540 \times 10^{-65}$$

$$Delta\_r := 1.519293907 \times 10^{22}$$

$$Delta\_p := 1.699029597 \times 10^{-57}$$

$$Delta\_E := 5.093562590 \times 10^{-49}$$

$$Delta\_t := 1.035198823 \times 10^{14}$$

$$\hbar 2 := 5.272850000 \times 10^{-35}$$

$$UP\_rm := 2.581325315 \times 10^{-35}$$

$$UP\_Et := 5.272849998 \times 10^{-35}$$

$$sigma\_r := 2.194469124 \times 10^{22}$$

$$delta\_min := 3.103447997 \times 10^{22}$$

```
R       = 3.922800e+22  m
GM      = 6.674000e+42  m^3/s^2
c2      = 2.449149e+15  m^3/s^2
N_wf    = 7.881727e-35  m^(-3/2)
k1      = 3.222223e-23  m^-1
omega1  = 9.659980e-15  rad/s
T1      = 6.504346e+14  s   = 20.609 Myr
E1      = 1.018713e-48  J
m_g    <= 1.133471e-65  kg
Delta_r= 1.519294e+22  m   = 0.387298 R
Delta_p= 1.699030e-57  kg m/s
sigma_r= 2.194469e+22  m   = 0.5594 R
```

```
================================================================
 SECTION 2: FORCE TABLE  F_mod vs F_Newton
================================================================
```

$$r\_test\_list := \left[1. \times 10^{10}, 1. \times 10^{15}, 1. \times 10^{18}, 1. \times 10^{20}, 1. \times 10^{21}, 3.9228 \times 10^{22}, 5.88420 \times 10^{22}, 7.84560 \times 10^{22}\right]$$

r_lbl_list := ["1e10", "1e15", "1e18", "1e20", "1e21", "R", "1.5R", "2R"]

```
Scale         F_mod (N/kg)  F_Newton (N/kg)  delta=r/(2R)
---------------------------------------------------------
```

$$rv := 1. \times 10^{10}$$

$$Fmod := -6.674000000 \times 10^{22}$$

$$FNewt := -6.674000000 \times 10^{22}$$

$$\delta := 1.274599776 \times 10^{-13}$$

```
1e10          -6.6740e+22      -6.6740e+22      0.000000
```

$$rv := 1. \times 10^{15}$$

$$Fmod := -6.673999915 \times 10^{12}$$

$$FNewt := -6.674000000 \times 10^{12}$$

$$\delta := 1.274599776 \times 10^{-8}$$

```
1e15          -6.6740e+12      -6.6740e+12      0.000000
```

$$rv := 1. \times 10^{18}$$

$$Fmod := -6.673914935 \times 10^{6}$$

$$FNewt := -6.674000000 \times 10^{6}$$

$$\delta := 0.00001274599776$$

```
1e18          -6.6739e+06      -6.6740e+06      0.000013
```

$$rv := 1. \times 10^{20}$$

$$Fmod := -666.5493325$$

$$FNewt := -667.4000000$$

$$\delta := 0.001274599776$$

```
1e20          -6.6655e+02      -6.6740e+02      0.001275
```

$$rv := 1. \times 10^{21}$$

$$Fmod := -6.588933215$$

$$FNewt := -6.674000000$$

$$\delta := 0.01274599776$$

```
1e21          -6.5889e+00      -6.6740e+00      0.012746
```

$$rv := 3.9228 \times 10^{22}$$

$$Fmod := -0.002168522205$$

$$FNewt := -0.004337044408$$

$$\delta := 0.5000000000$$

```
R             -2.1685e-03      -4.3370e-03      0.500000
```

$$rv := 5.88420 \times 10^{22}$$

$$Fmod := -0.0004818938233$$

$$FNewt := -0.001927575293$$

$$\delta := 0.7500000000$$

```
1.5R          -4.8189e-04      -1.9276e-03      0.750000
```

$$rv := 7.84560 \times 10^{22}$$

$$Fmod := 0.$$

$$FNewt := -0.001084261102$$

$$\delta := 1.000000000$$

```
  2R              0.0000e+00      -1.0843e-03      1.000000

==============================================================
 SECTION 2b: HEISENBERG PRODUCTS
==============================================================
  Delta_r * Delta_p = 2.581325e-35  J s
  hbar/2            = 5.272850e-35  J s
  Ratio pos-mom     = 0.489550  (linear psi; Airy exact = 1.082)
  Delta_E * Delta_t = 5.272850e-35  J s
  Ratio E-t         = 1.000000  (saturated exactly = 1.0)

==============================================================
 SECTION 3: AIRY MODE TABLE
==============================================================
```

$$airy\_z := [-2.33810741, -4.08794944, -5.52055983, -6.78670809, -7.94413359]$$

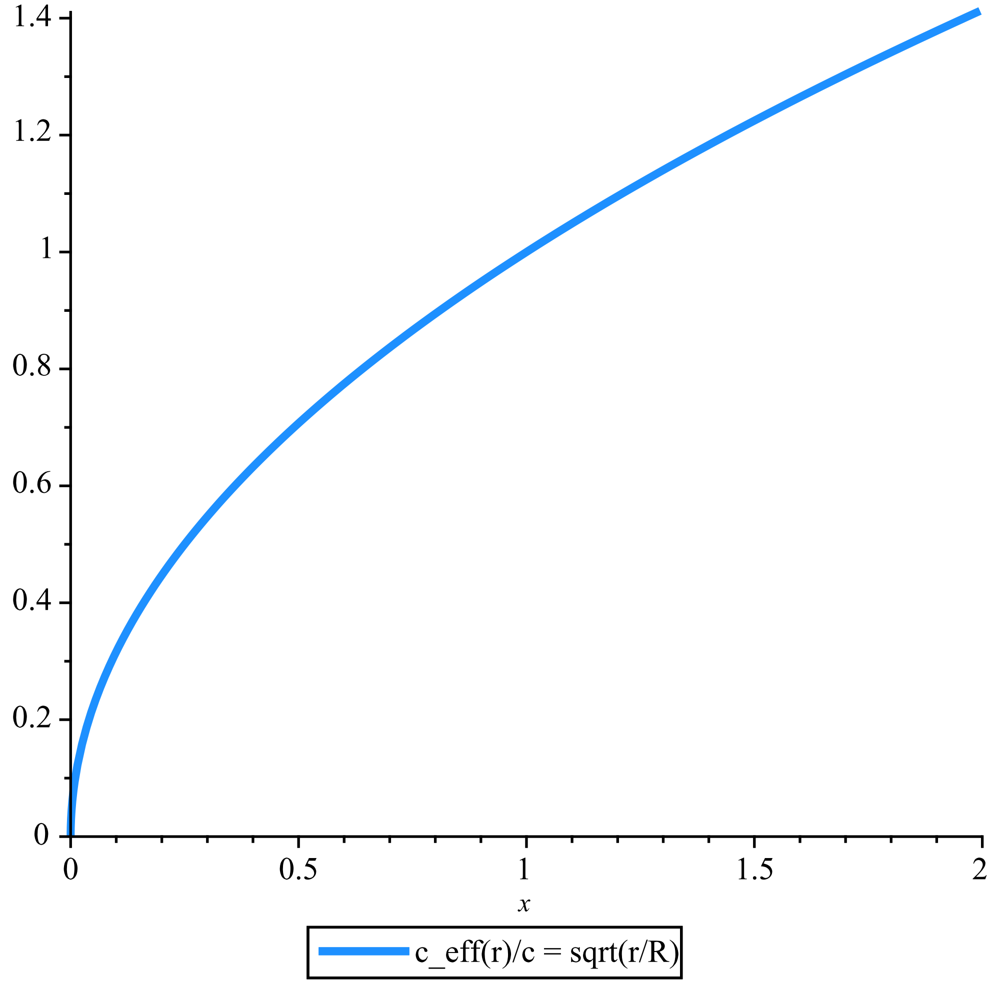
$$kn\_list := [3.222222512 \times 10^{-23}, 7.449328282 \times 10^{-23}, 1.169051172 \times 10^{-22}, 1.593483586 \times 10^{-22}, 2.018034783 \times 10^{-22}]$$

$$En\_list := [1.018712518 \times 10^{-48}, 2.355121021 \times 10^{-48}, 3.695980208 \times 10^{-48}, 5.037832336 \times 10^{-48}, 6.380059999 \times 10^{-48}]$$

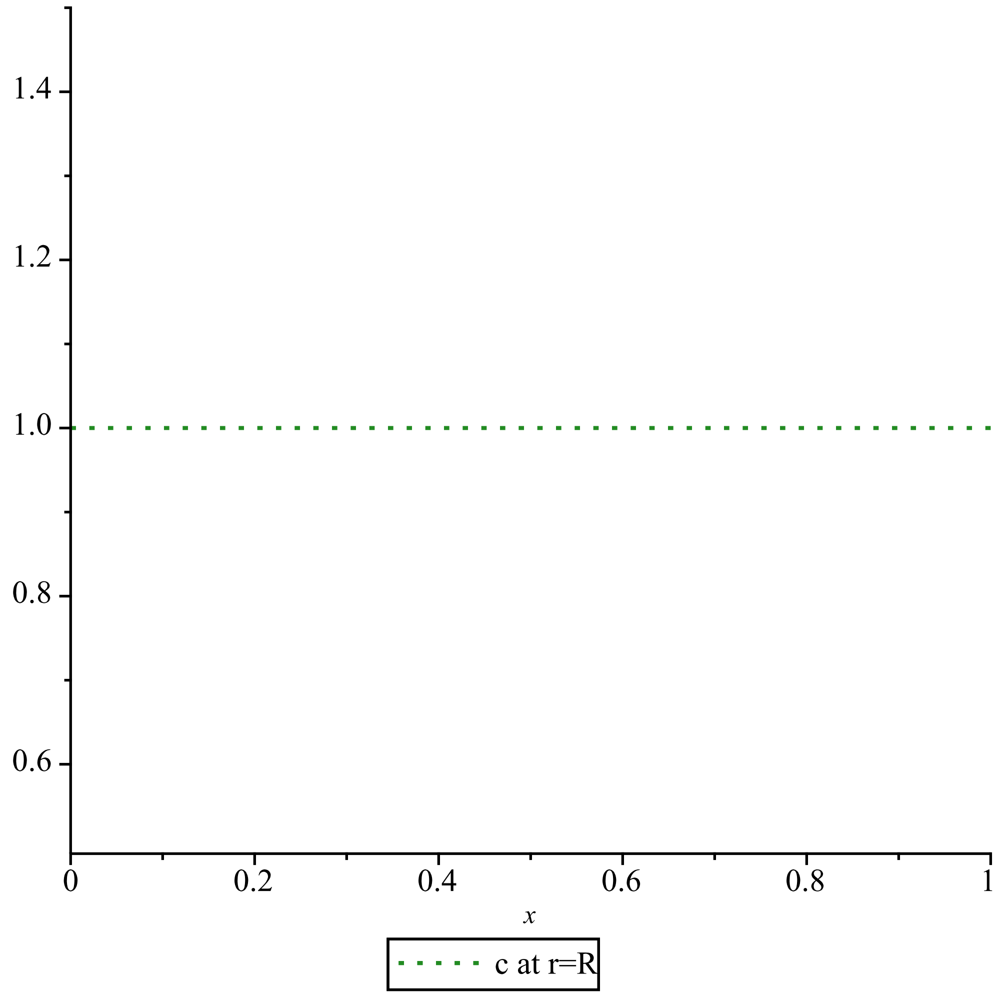
$$En\_norm := [1.000000000, 2.311860294, 3.628089518, 4.945293443, 6.262866006]$$

$$Tn\_Myr := [20.60946146, 8.914665615, 5.680527275, 4.167490098, 3.290739646]$$

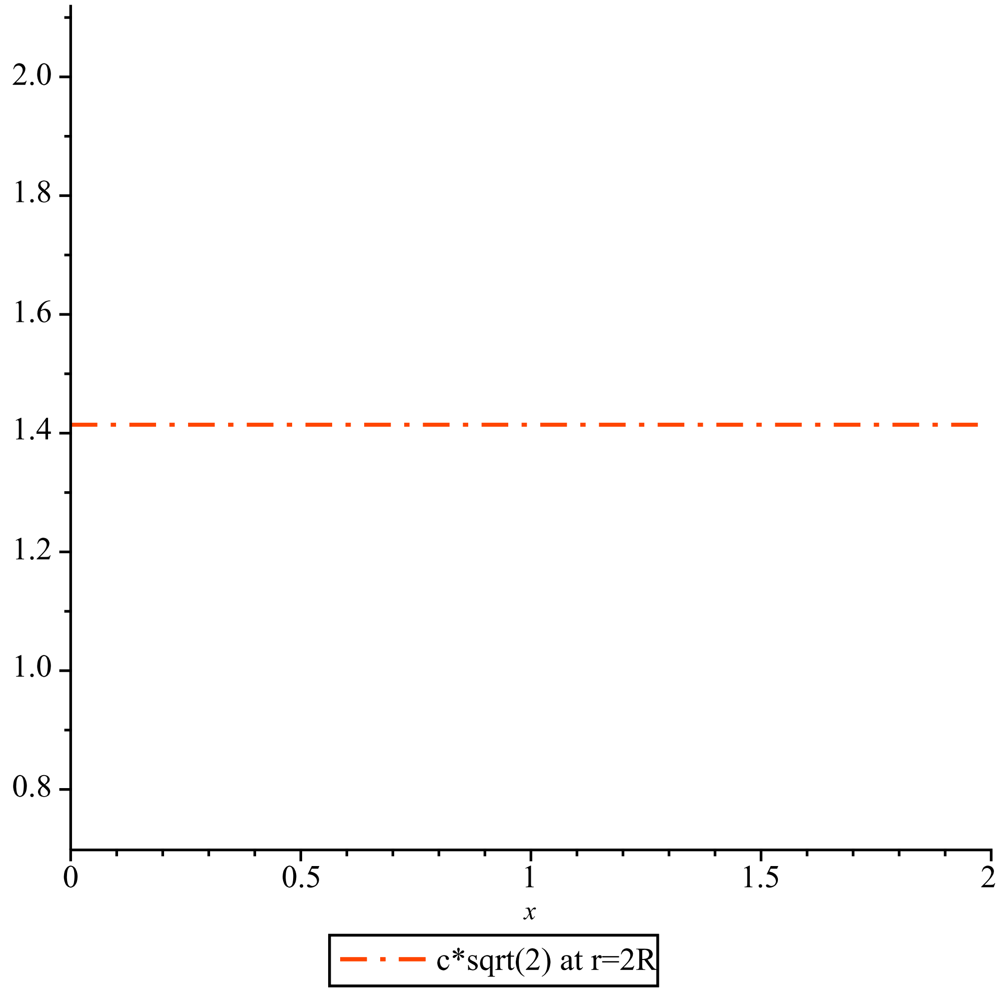
```
  n     a_n           k_n (m^-1)      E_n/E_1       T_n (Myr)
  ----------------------------------------------------------
  1     -2.338107     3.2222e-23      1.000000      20.609
  2     -4.087949     7.4493e-23      2.311860      8.915
  3     -5.520560     1.1691e-22      3.628090      5.681
  4     -6.786708     1.5935e-22      4.945293      4.167
  5     -7.944134     2.0180e-22      6.262866      3.291

==============================================================
 PLOT 01: Effective Propagation Speed c_eff(r)/c
==============================================================
```

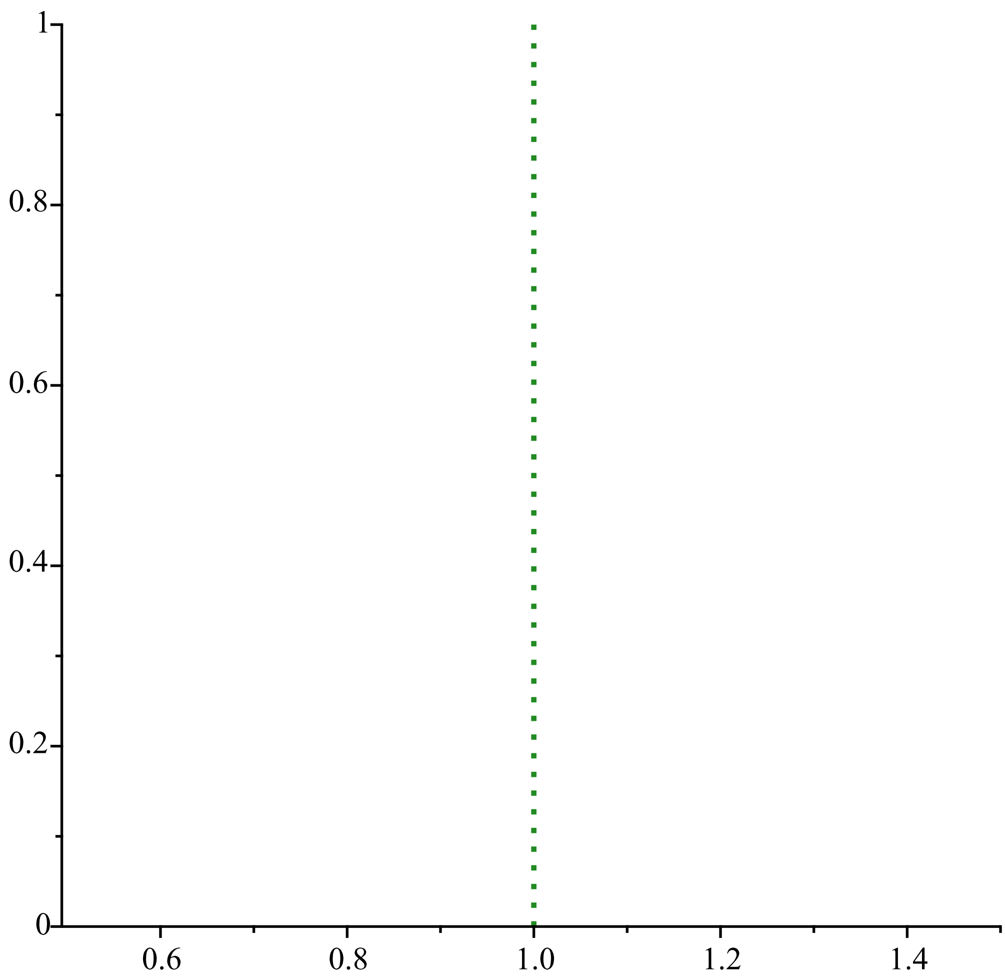


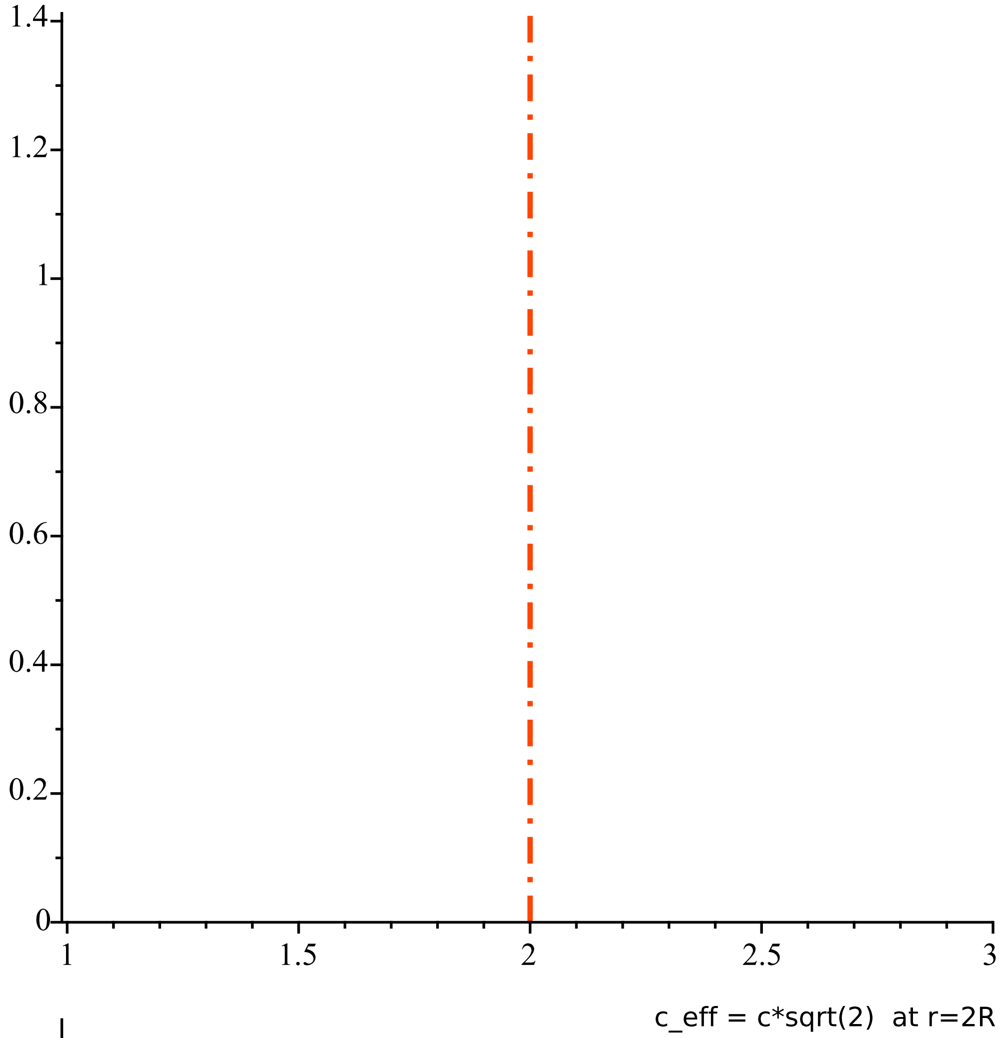
x
c_eff(r)/c = sqrt(r/R)

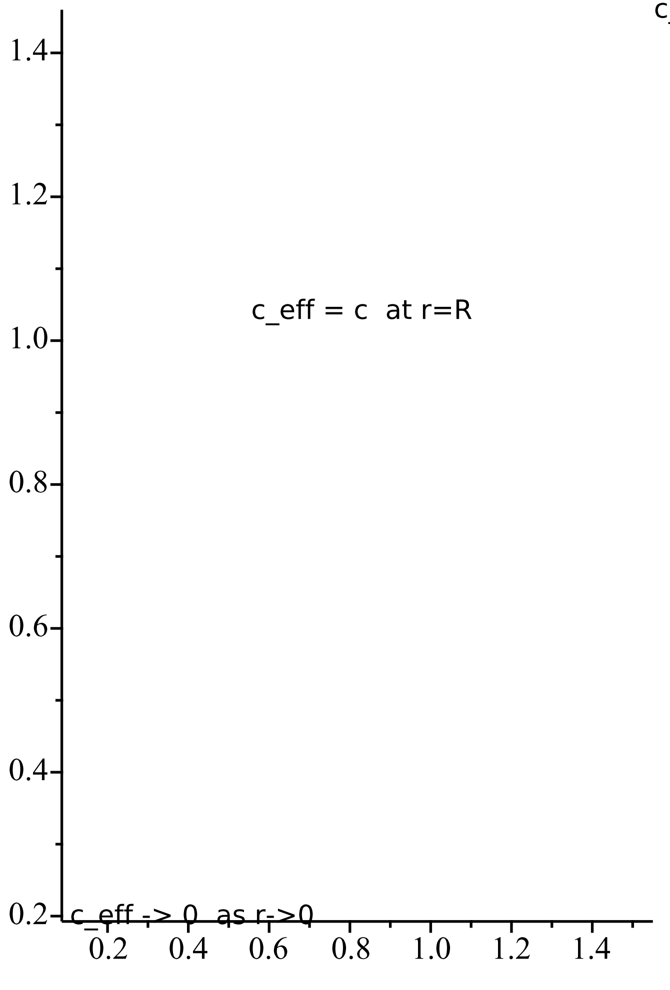


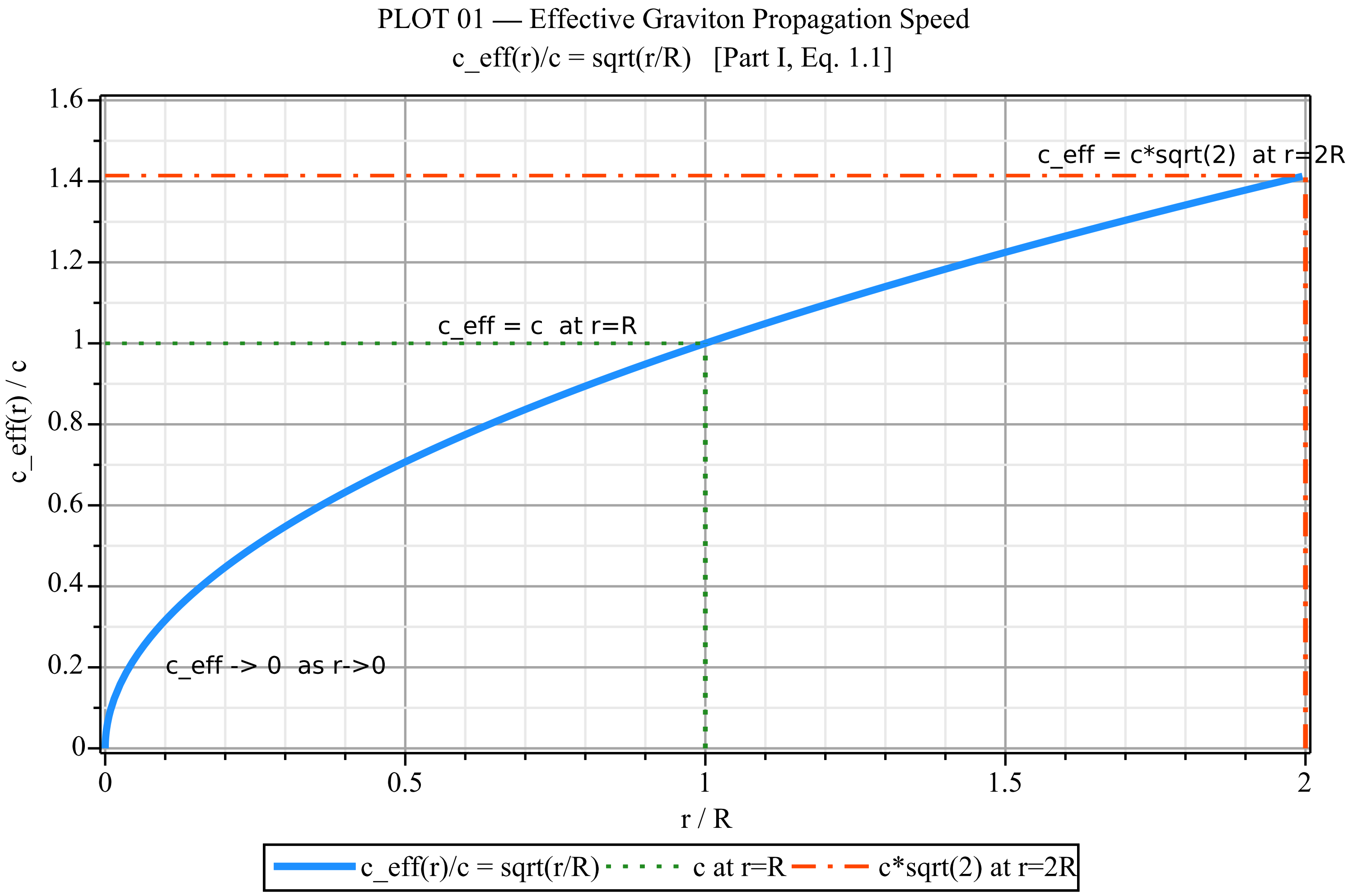
x
c at r=R


2.0
1.8
1.6
1.4
1.2
1.0
0.8
0
0.5
1
1.5
2
x
c*sqrt(2) at r=2R


1
0.8
0.6
0.4
0.2
0
0.6
0.8
1.0
1.2
1.4

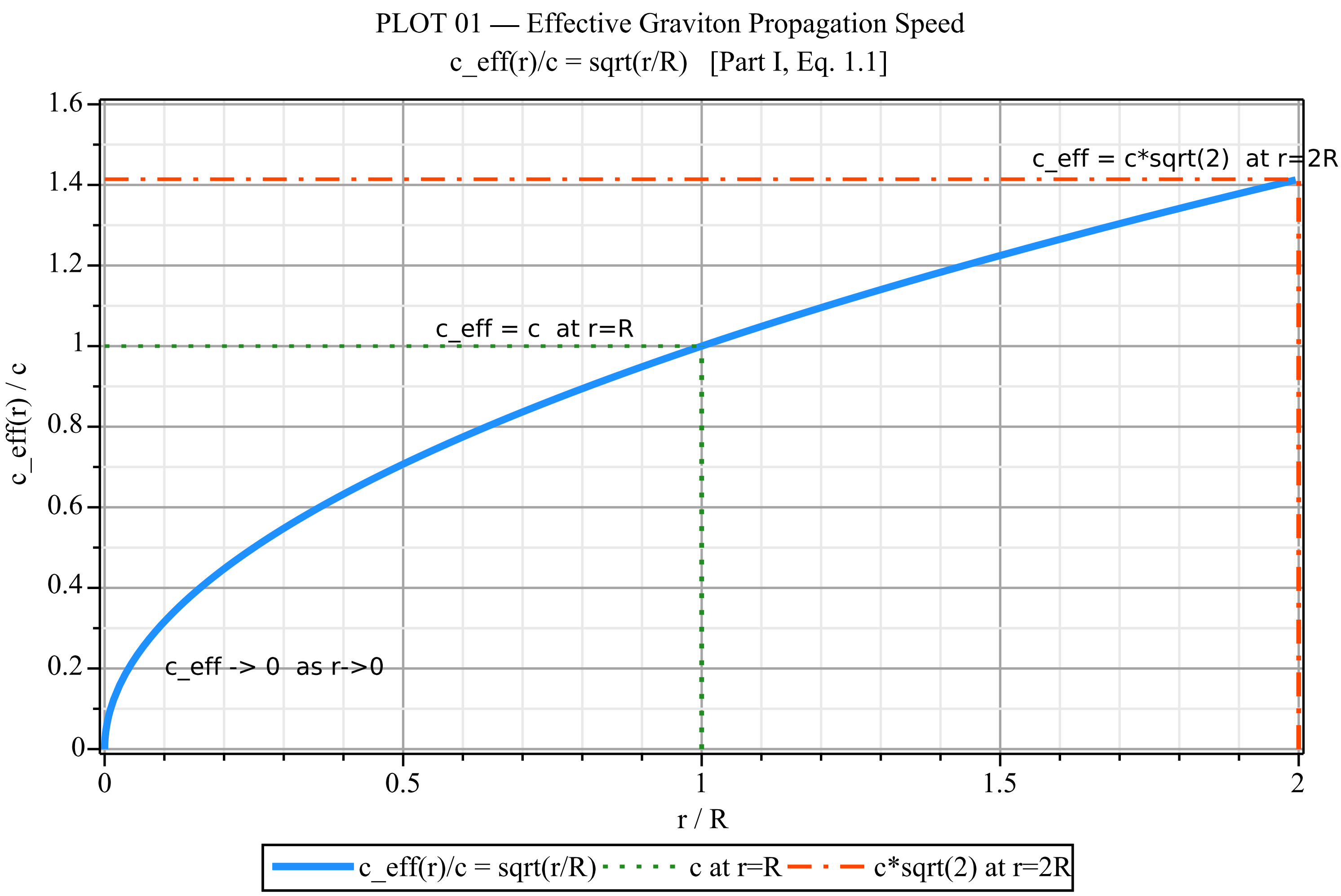

1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
1.5
2
2.5
3
c_eff = c*sqrt(2) at r=2R


1.4
1.2
1.0
0.8
0.6
0.4
0.2
c_eff = c at r=R
c_eff -> 0 as r->0
0.2
0.4
0.6
0.8
1.0
1.2
1.4



PLOT 01 — Effective Graviton Propagation Speed
c_eff(r)/c = sqrt(r/R)   [Part I, Eq. 1.1]
c_eff = c*sqrt(2) at r=2R
c_eff = c at r=R
c_eff -> 0 as r->0
c_eff(r) / c
r / R
c_eff(r)/c = sqrt(r/R)
c at r=R
c*sqrt(2) at r=2R

"PLOT 01 done."



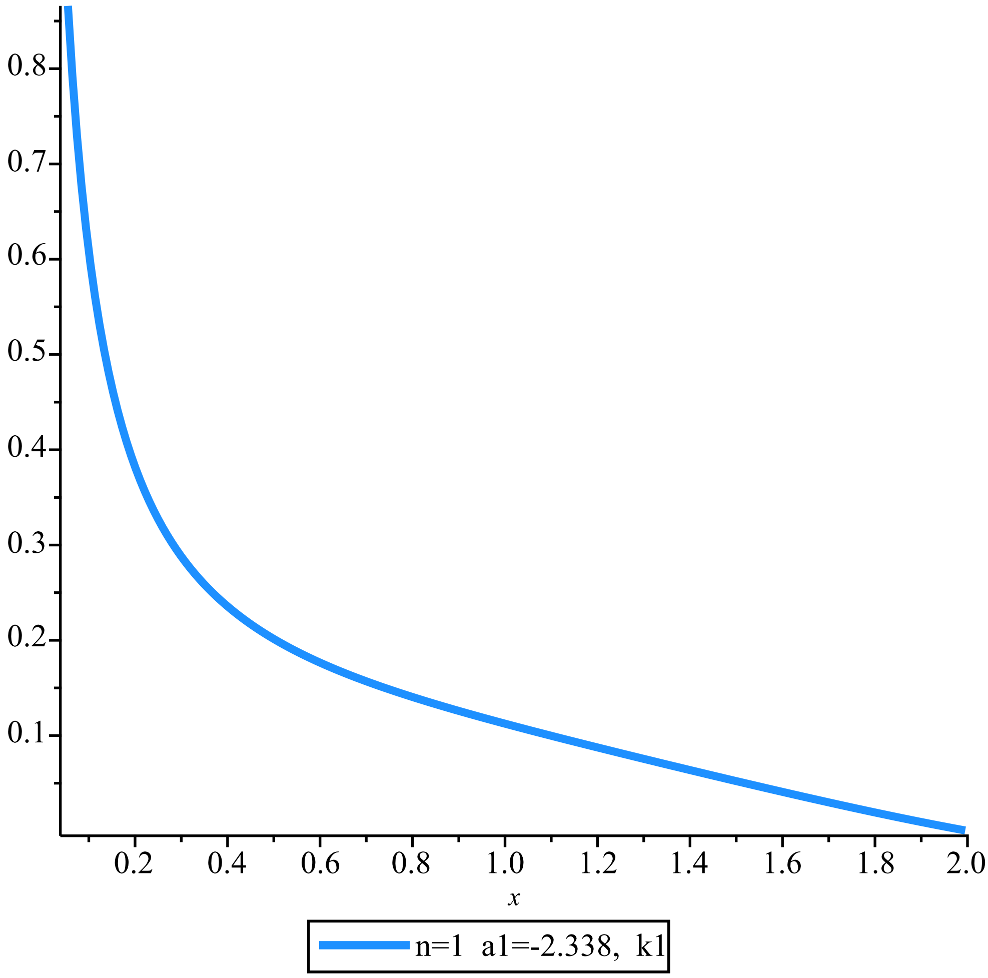

```
==============================================================
 PLOT 02: Airy Eigenstates g_n(r/R)
==============================================================
```

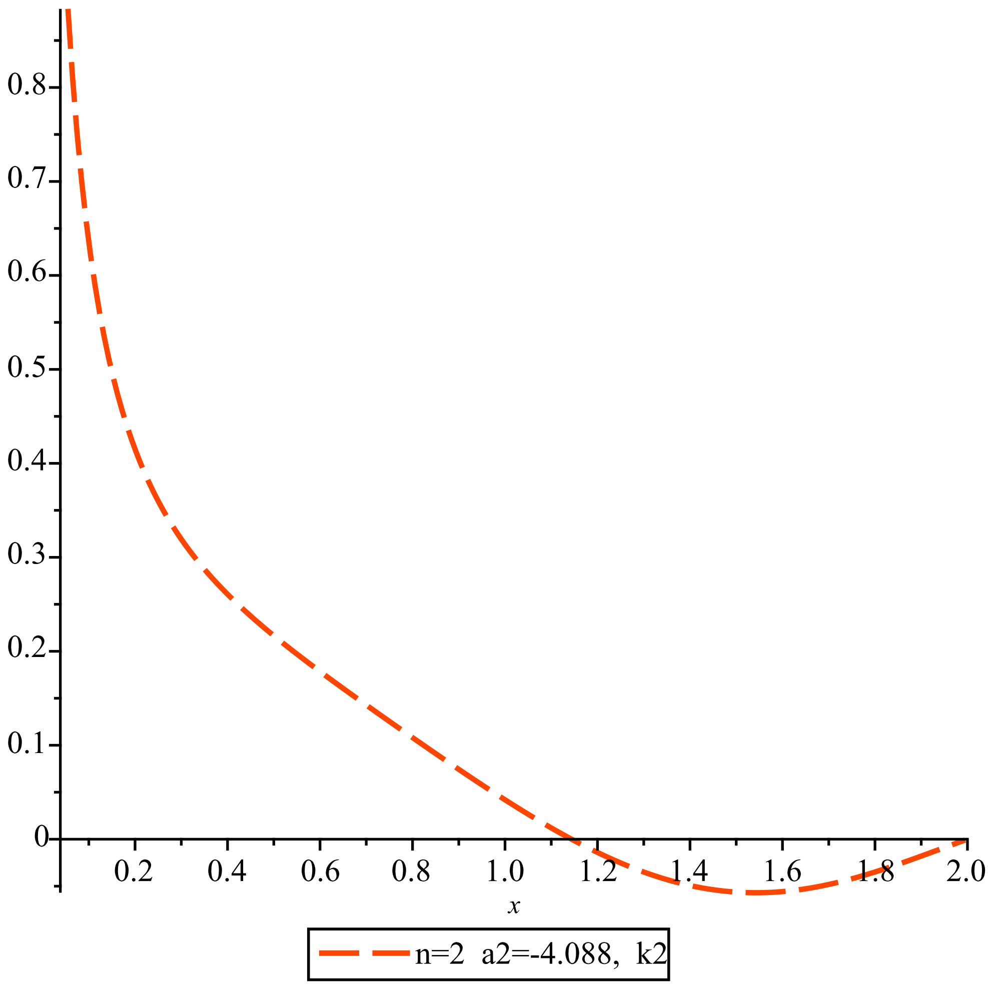
$$eps := 0.02$$

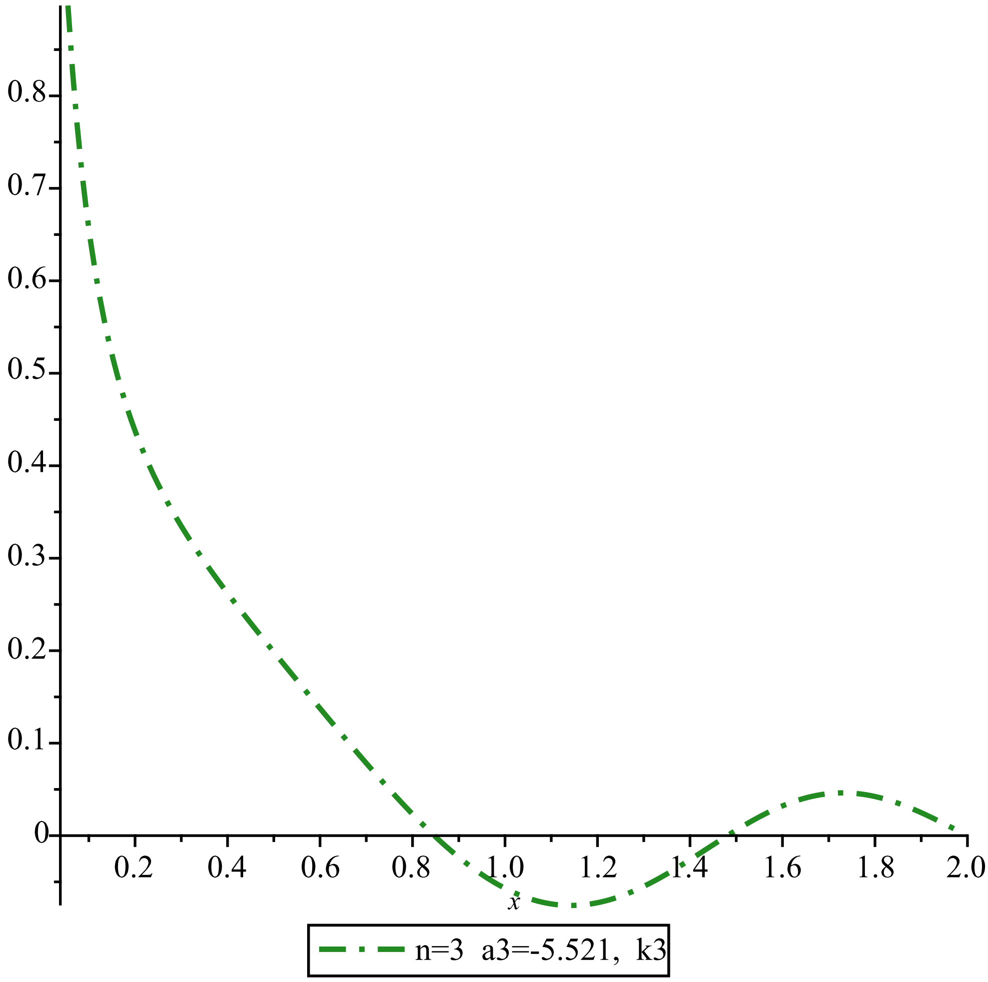
$$g1mx := 9.046969170$$

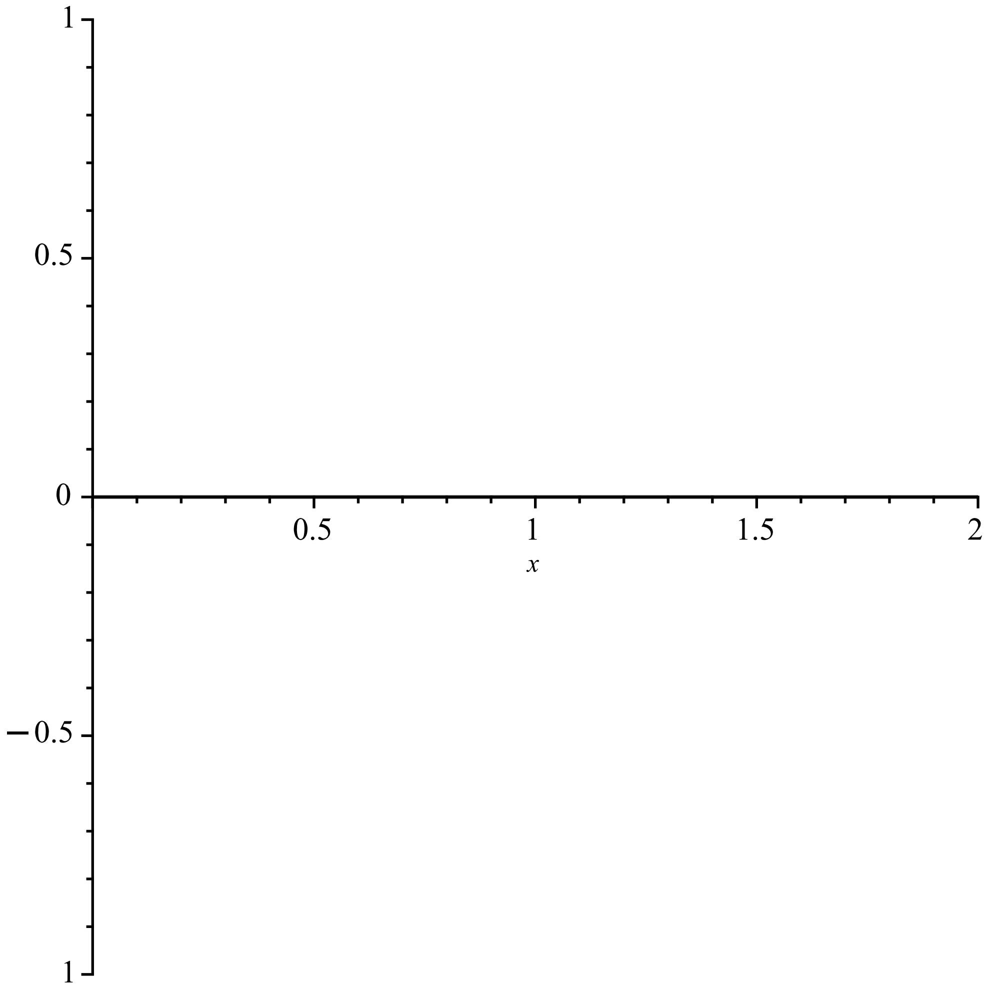
$$g2mx := 9.160108940$$

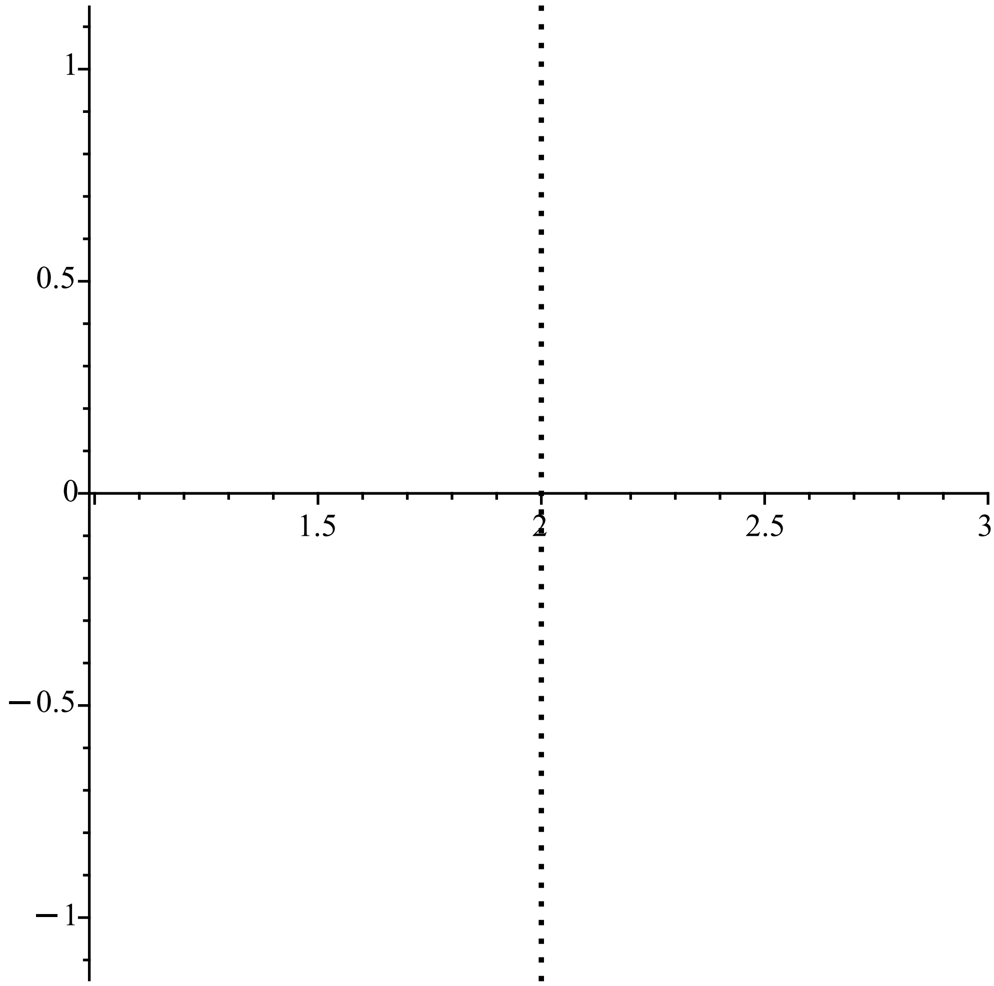
$$g3mx := 9.252654455$$

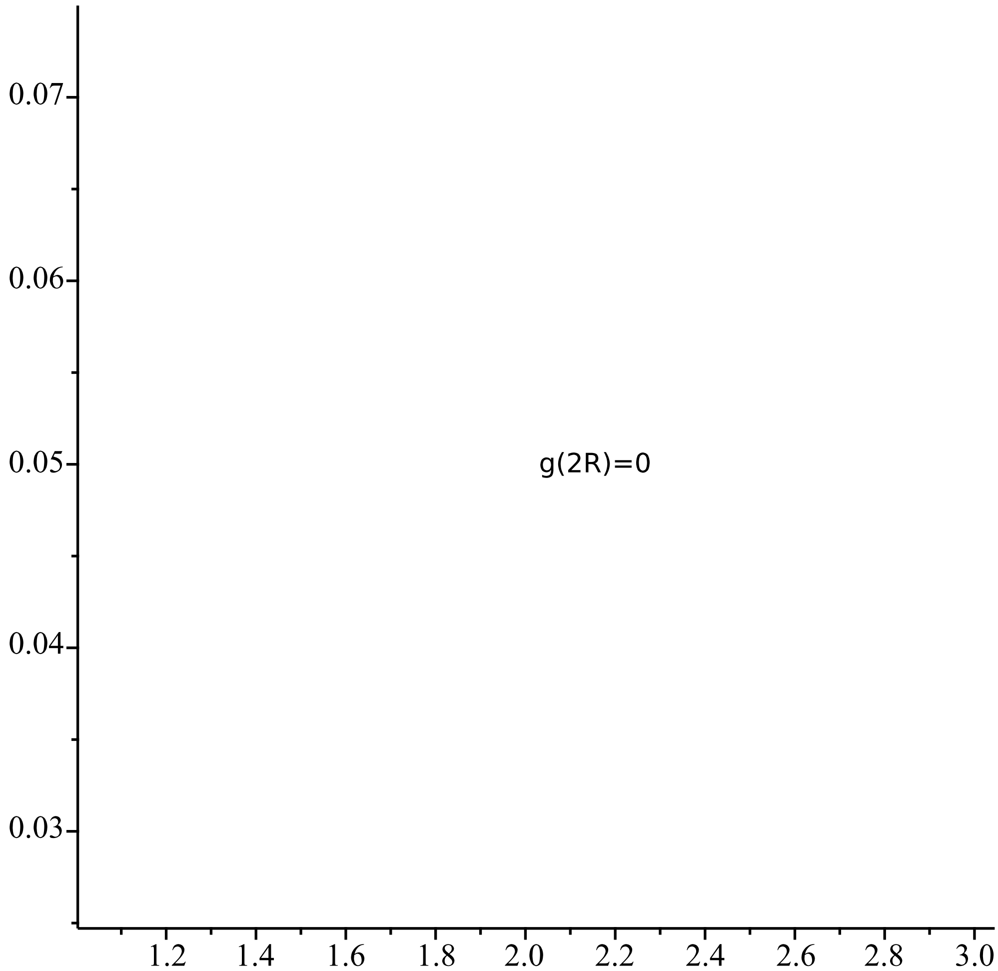

n=1 a1=-2.338, k1
x

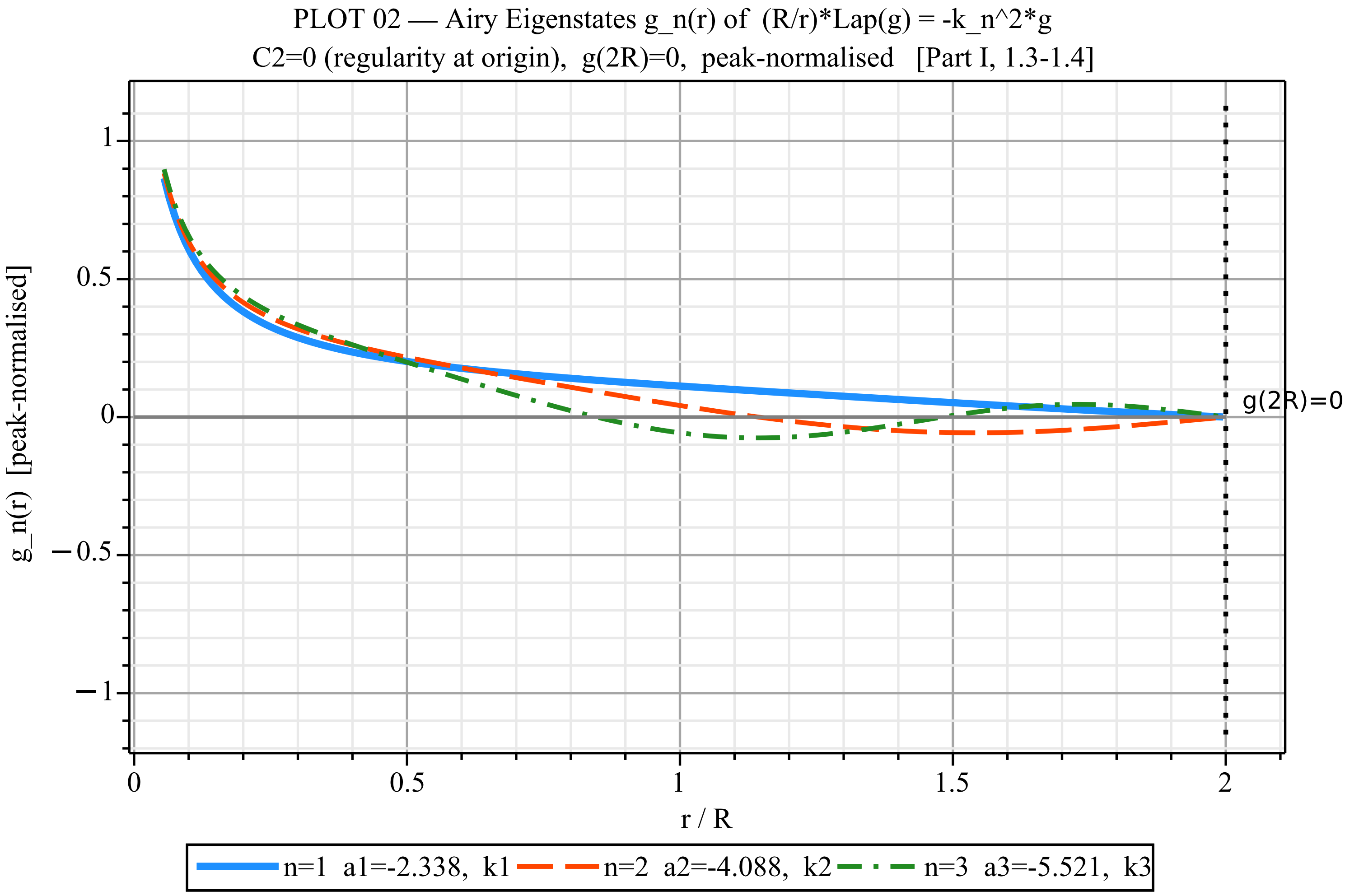

n=2 a2=-4.088, k2
x


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
x
n=3 a3=-5.521, k3


1
0.5
0
0.5
1
1.5
2
x
−0.5
1

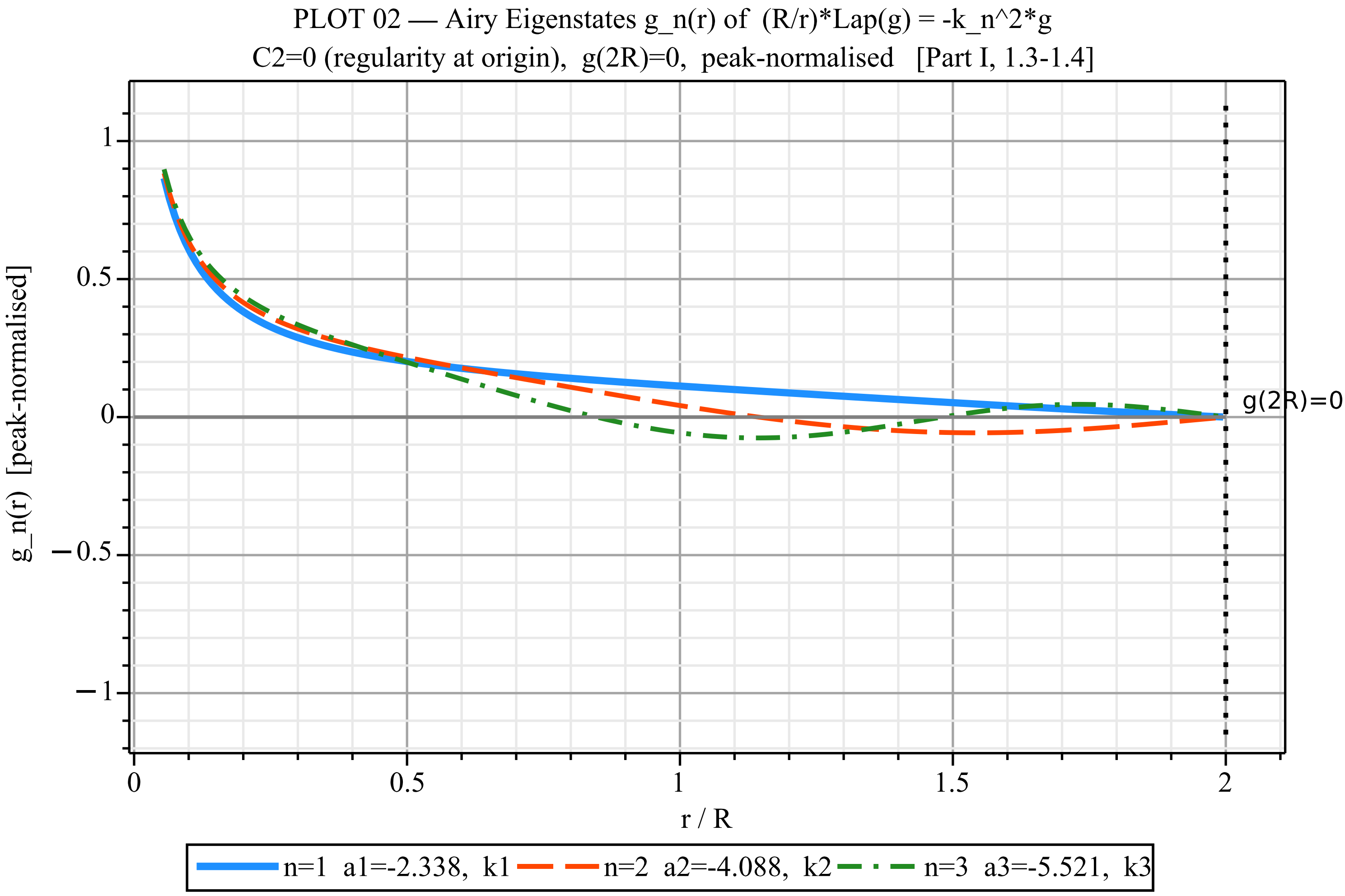

1
0.5
0
-0.5
-1
1.5
2
2.5
3


0.07
0.06
0.05
0.04
0.03
g(2R)=0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8
3.0

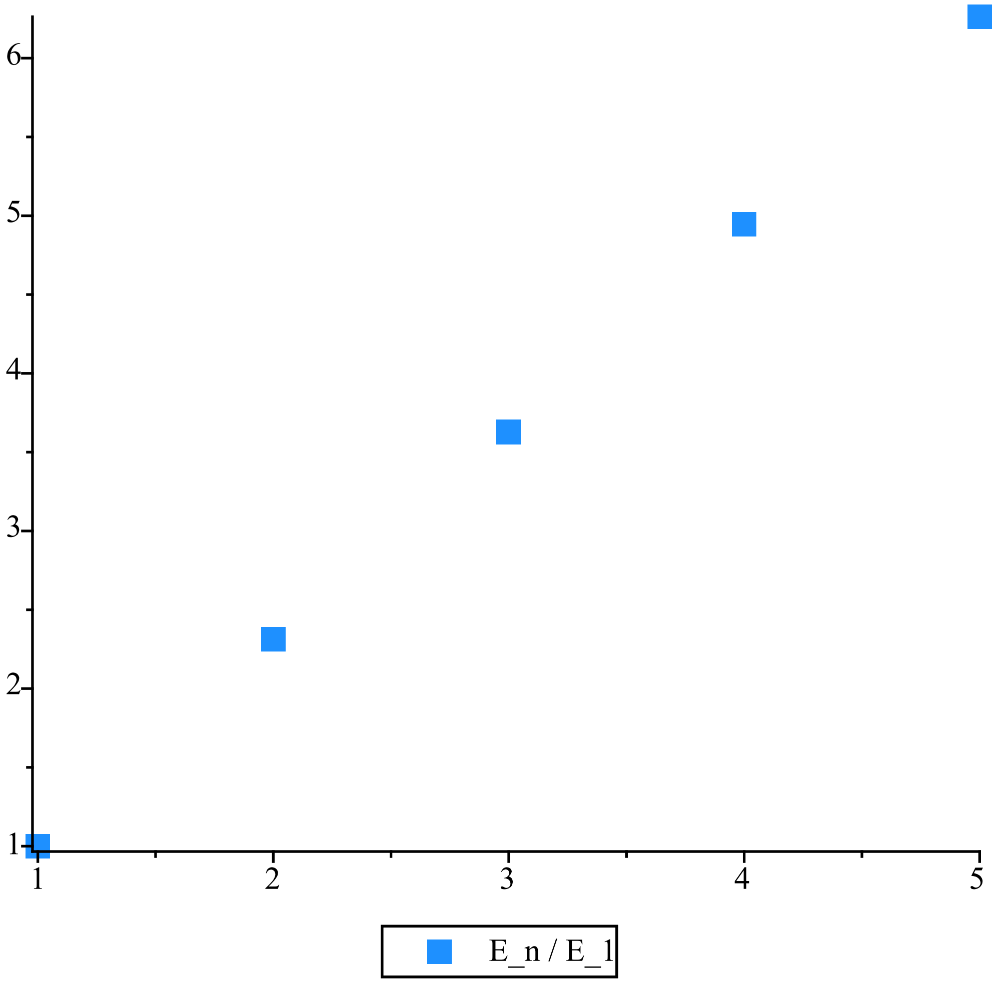


PLOT 02 — Airy Eigenstates g_n(r) of (R/r)*Lap(g) = -k_n^2*g
C2=0 (regularity at origin), g(2R)=0, peak-normalised [Part I, 1.3-1.4]
g_n(r) [peak-normalised]
1
0.5
0
−0.5
−1
0
0.5
1
1.5
2
r / R
g(2R)=0
n=1 a1=-2.338, k1
n=2 a2=-4.088, k2
n=3 a3=-5.521, k3

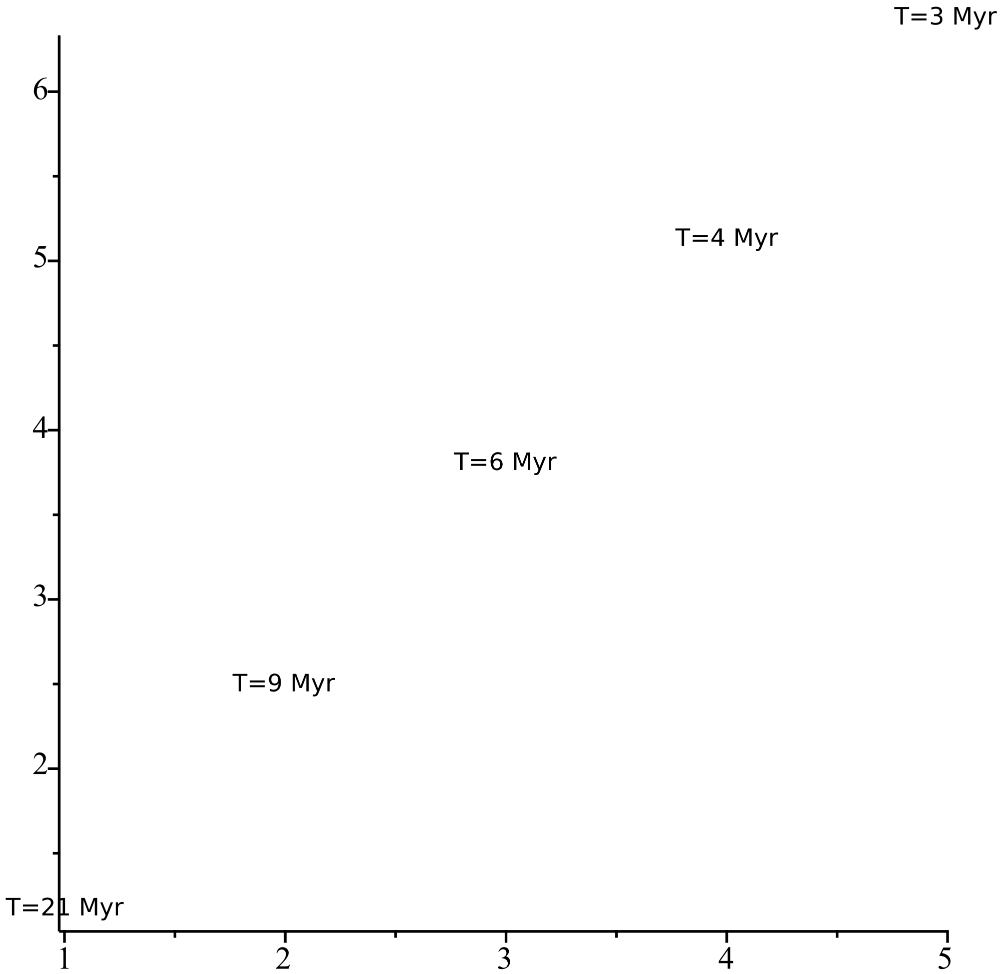
"PLOT 02 done."



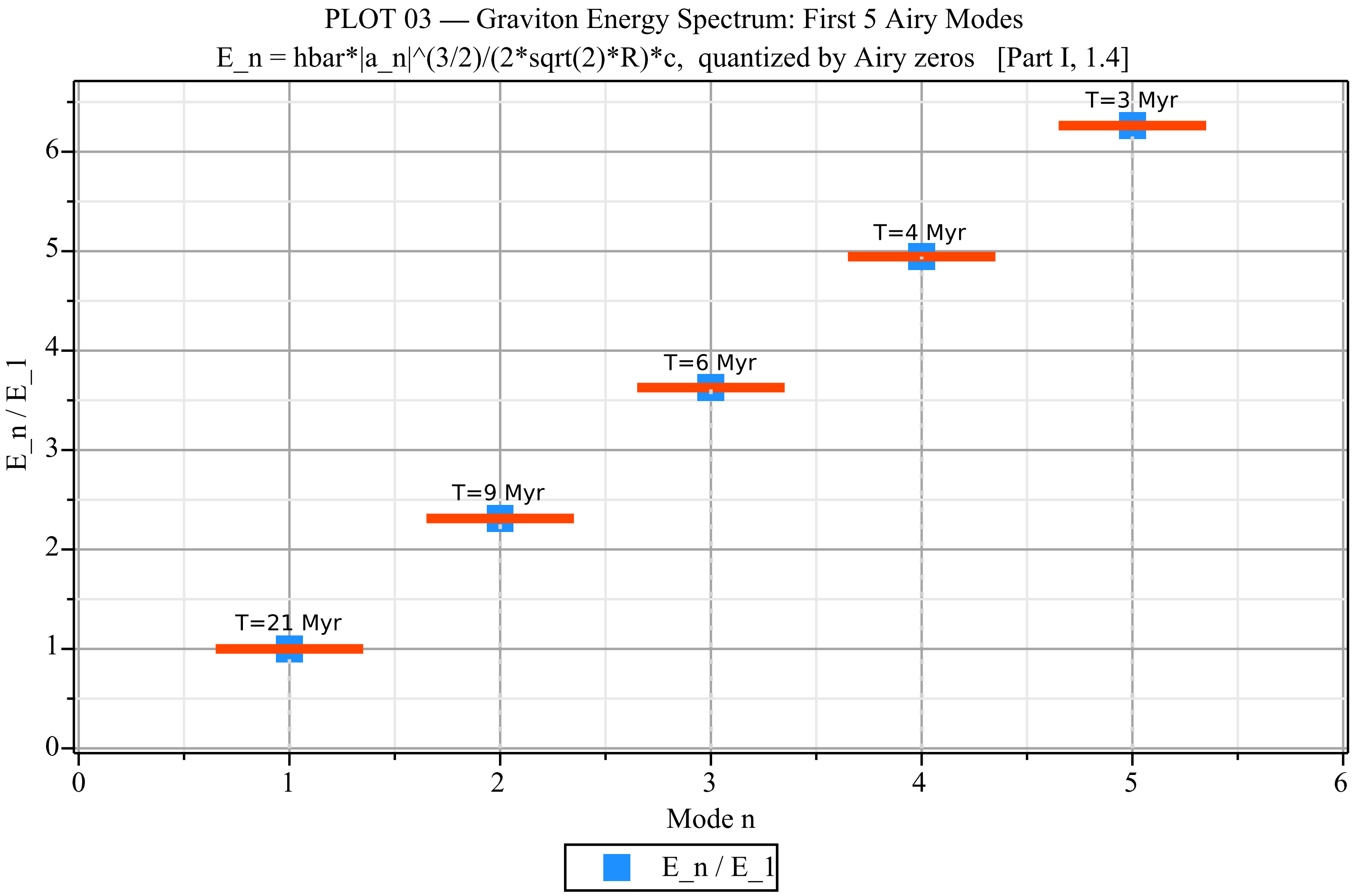

```
================================================================
 PLOT 03: Mode Spectrum E_n/E_1 (first 5 Airy modes)
================================================================
```



6
5
4
3
2
1
1
2
3
4
5
E_n / E_1



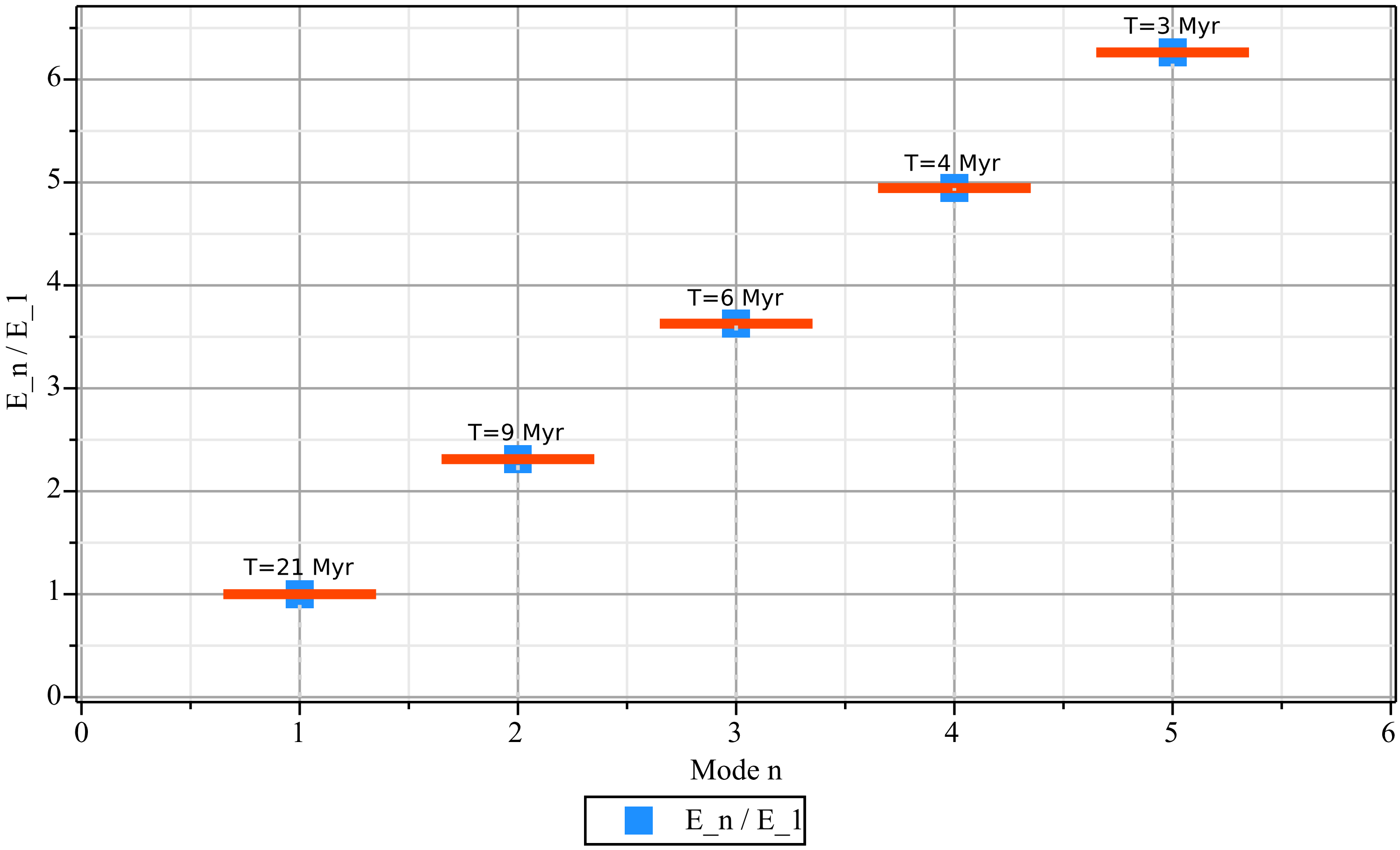
T=3 Myr
6
5
T=4 Myr
4
T=6 Myr
3
T=9 Myr
2
T=21 Myr
1
2
3
4
5



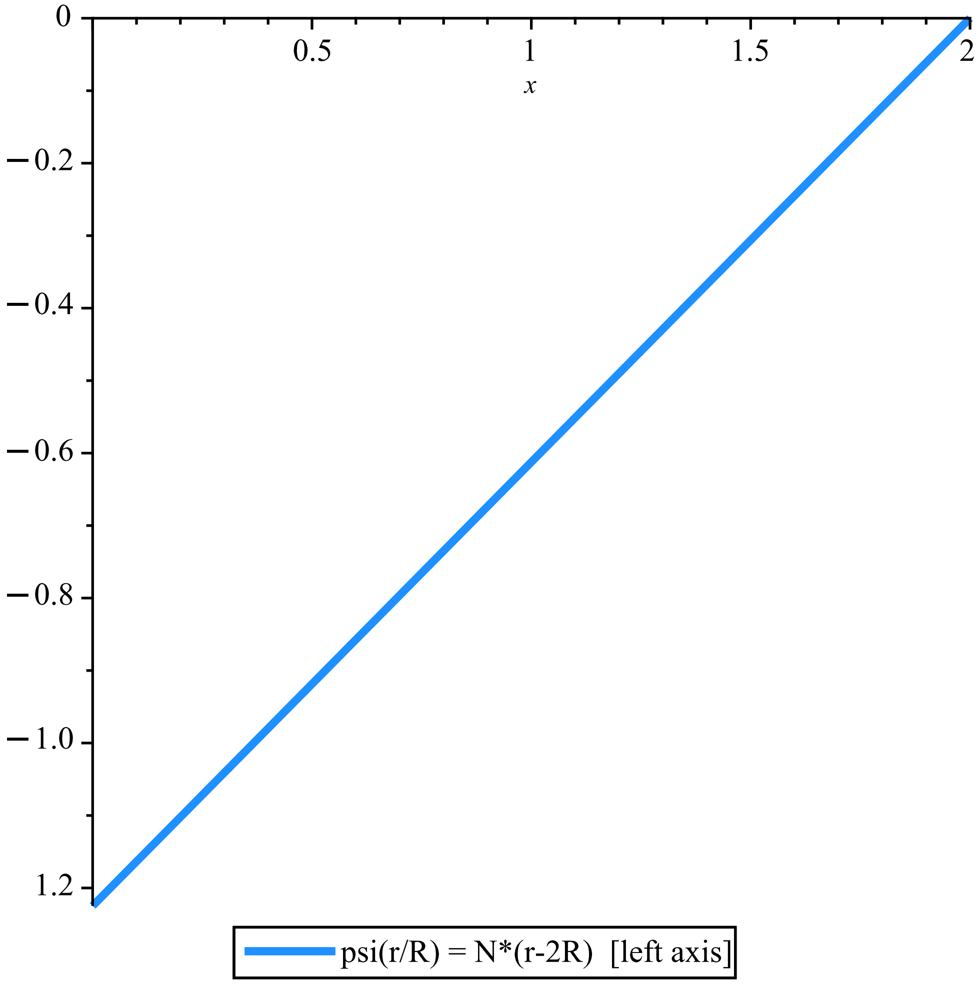
PLOT 03 — Graviton Energy Spectrum: First 5 Airy Modes
E_n = hbar*|a_n|^(3/2)/(2*sqrt(2)*R)*c, quantized by Airy zeros [Part I, 1.4]
T=21 Myr
T=9 Myr
T=6 Myr
T=4 Myr
T=3 Myr
E_n / E_1
Mode n
0
1
2
3
4
5
6
E_n / E_1

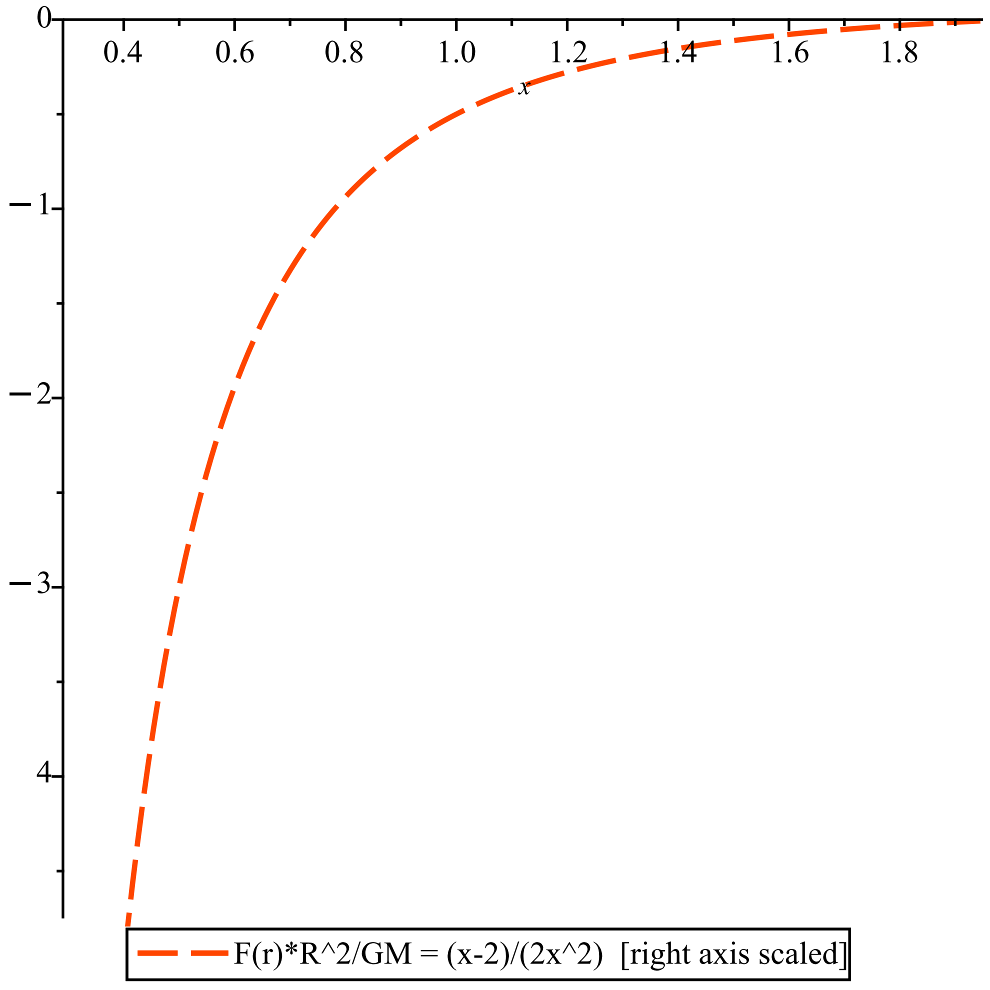
"PLOT 03 done."

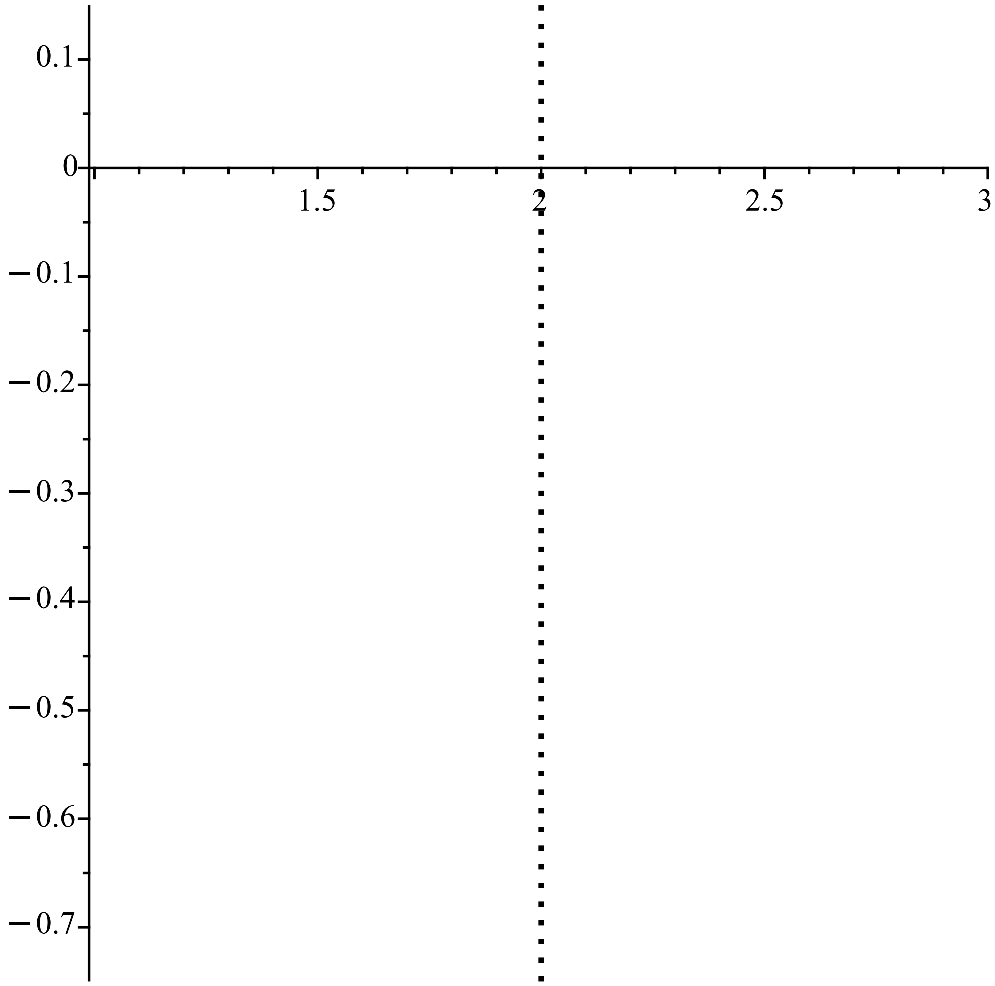
PLOT 03 — Graviton Energy Spectrum: First 5 Airy Modes
E_n = hbar*|a_n|^(3/2)/(2*sqrt(2)*R)*c, quantized by Airy zeros [Part I, 1.4]

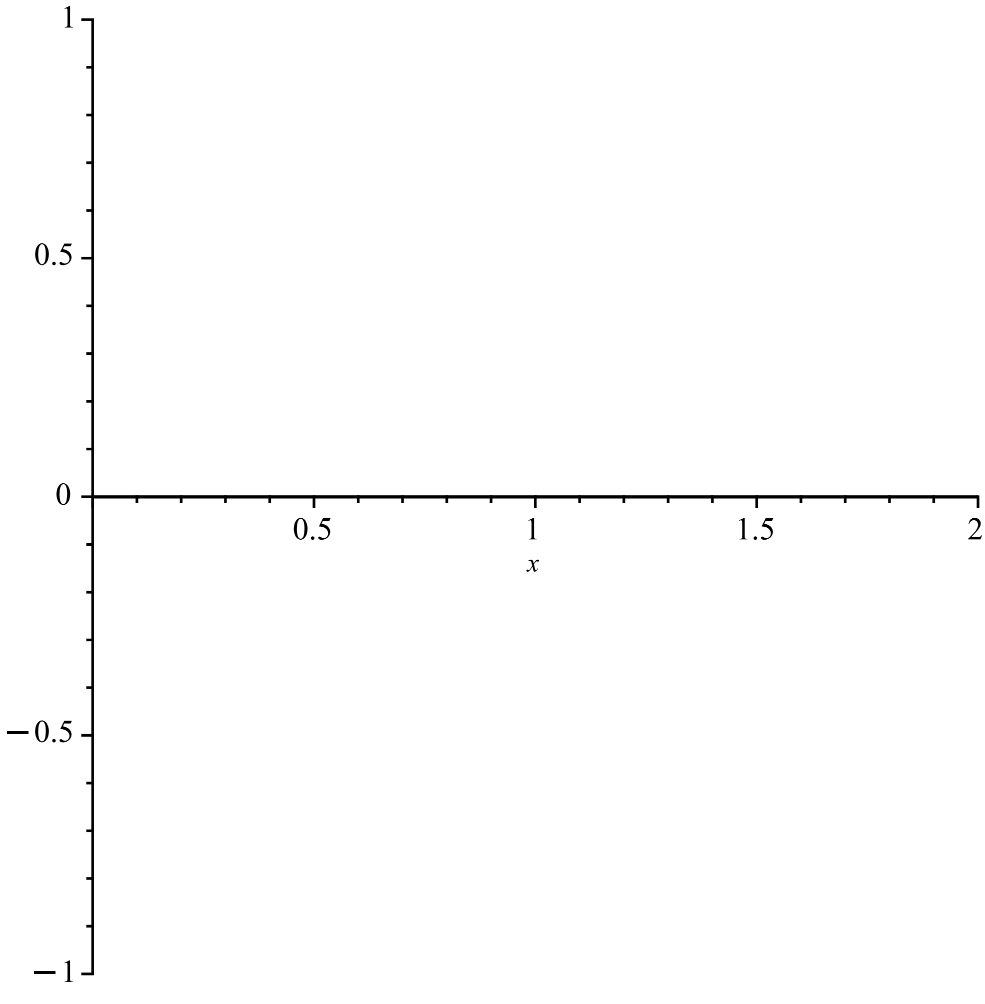


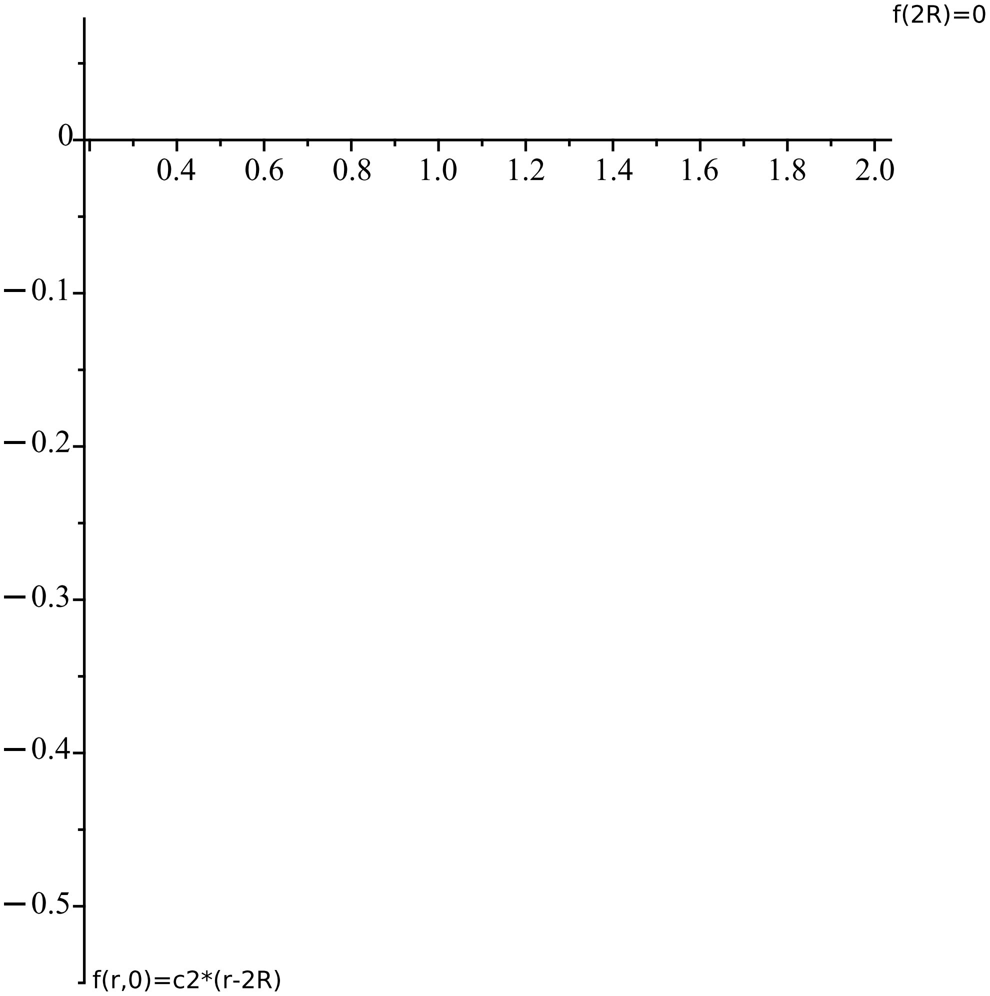
==
PLOT 04: Polynomial Wave Function and Graviton Flux
==


0
0.5
1
1.5
2
x
−0.2
−0.4
−0.6
−0.8
−1.0
1.2
psi(r/R) = N*(r-2R) [left axis]

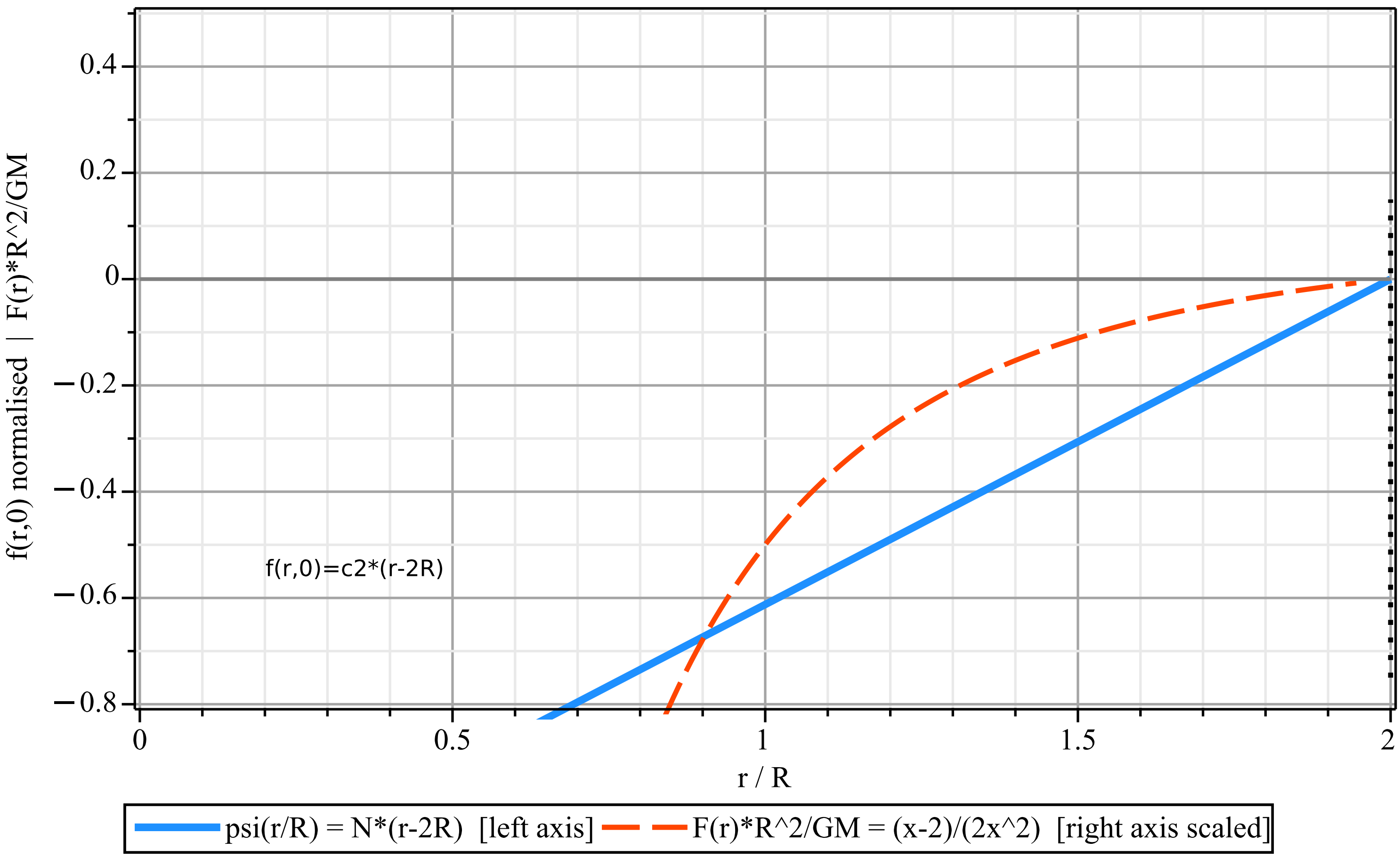

0
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
x
−1
−2
−3
4
F(r)*R^2/GM = (x-2)/(2x^2) [right axis scaled]


0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-0.6
-0.7
1.5
2
2.5
3


1
0.5
0
-0.5
-1
0.5
1
1.5
2
x

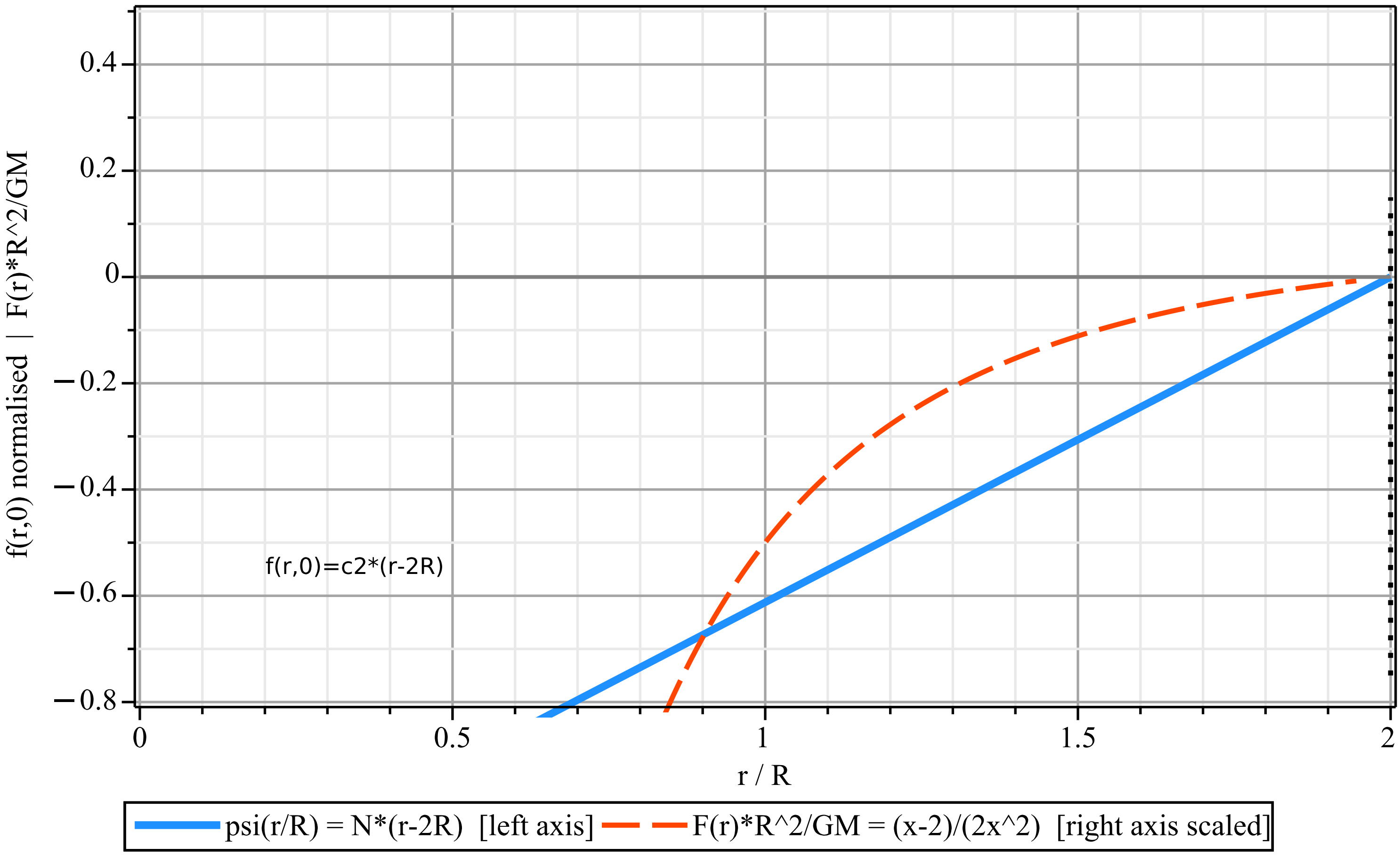


PLOT 04 — Polynomial Wave Function f(r,0) and Graviton Flux F(r,0)
f(x)=N*(x-2), F*R^2/GM=(x-2)/(2x^2), x=r/R [Part II, 2.1-2.2]

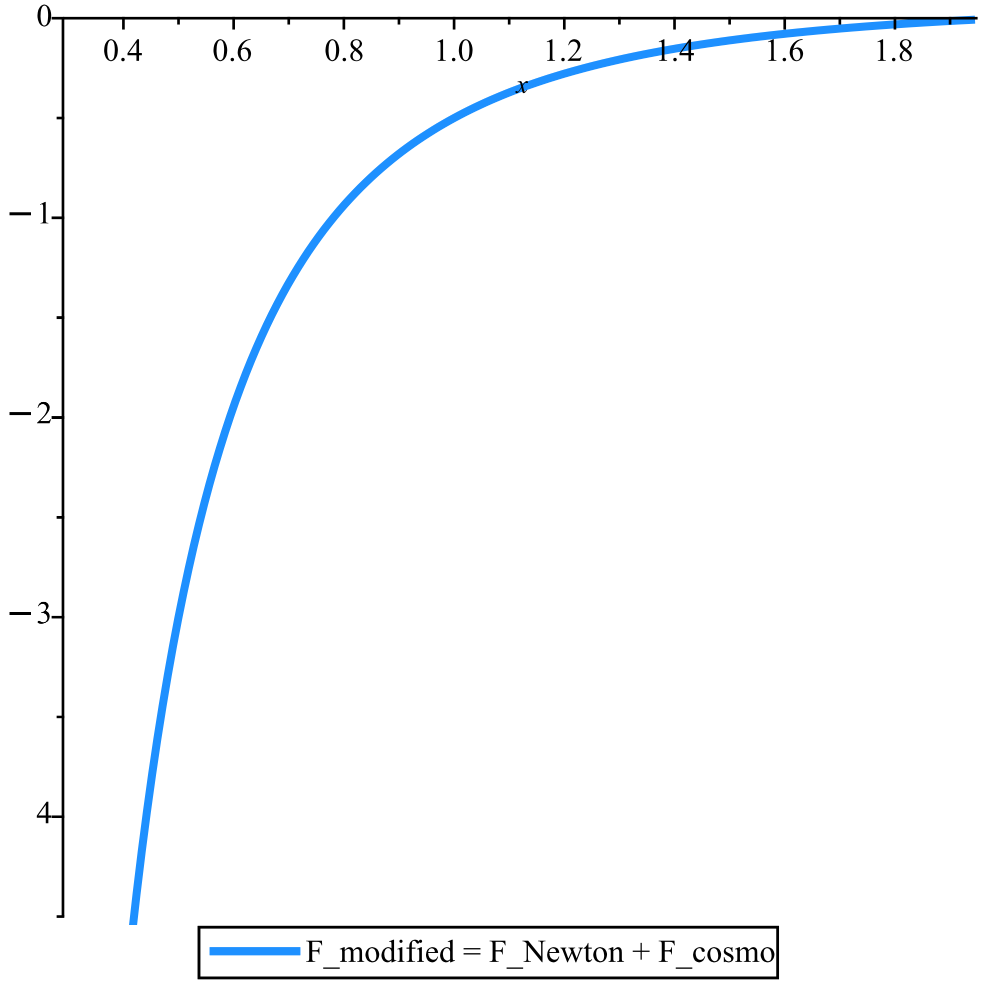


"PLOT 04 done."

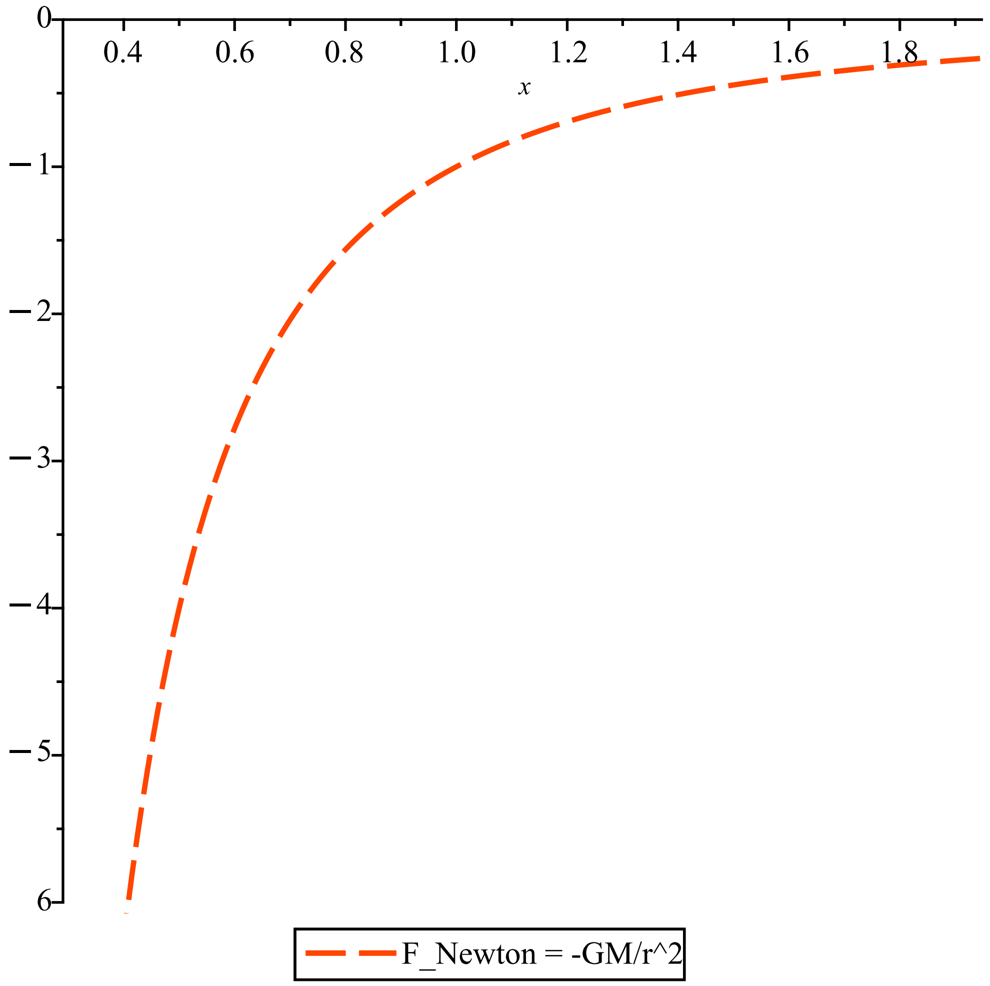
PLOT 04 — Polynomial Wave Function f(r,0) and Graviton Flux F(r,0)
f(x)=N*(x-2), F*R^2/GM=(x-2)/(2x^2), x=r/R [Part II, 2.1-2.2]



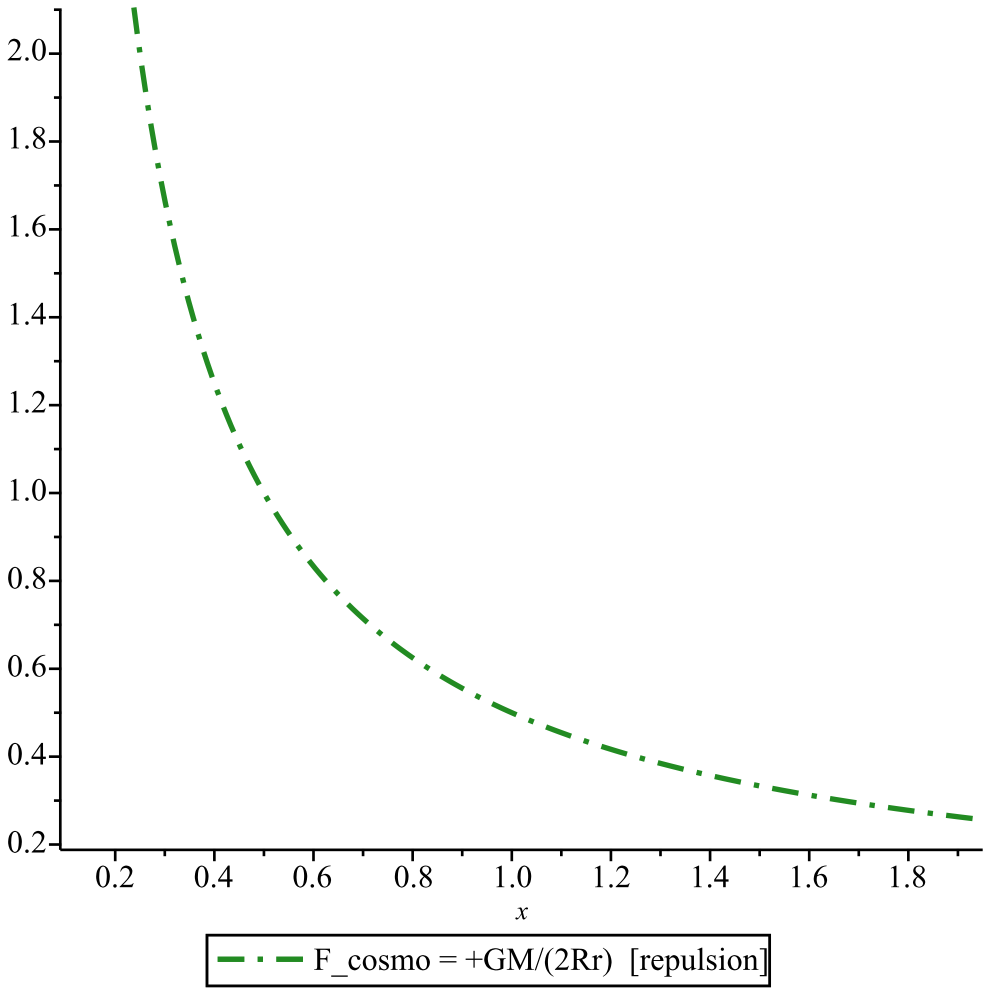

```
=============================================================
 PLOT 05: Force Decomposition F_mod = F_Newton + F_cosmo
=============================================================
```



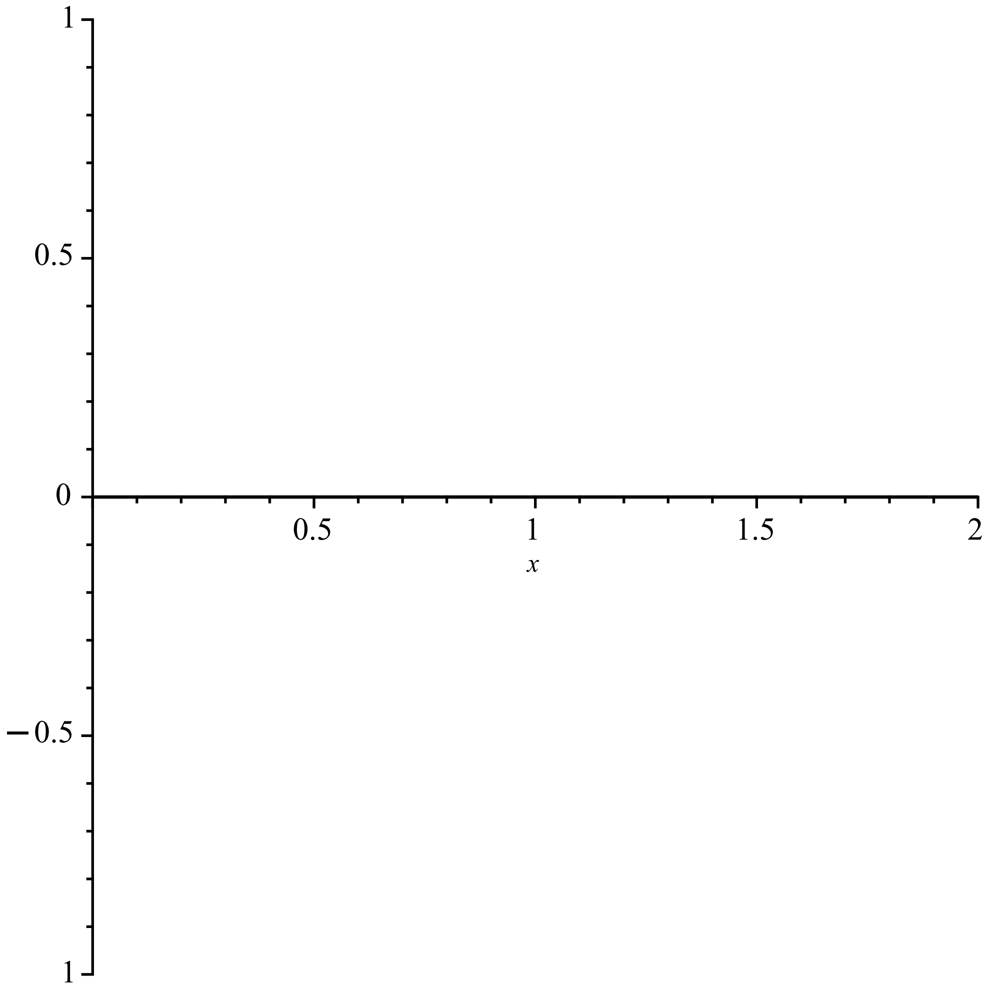
0
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
x
−1
−2
−3
4
F_modified = F_Newton + F_cosmo



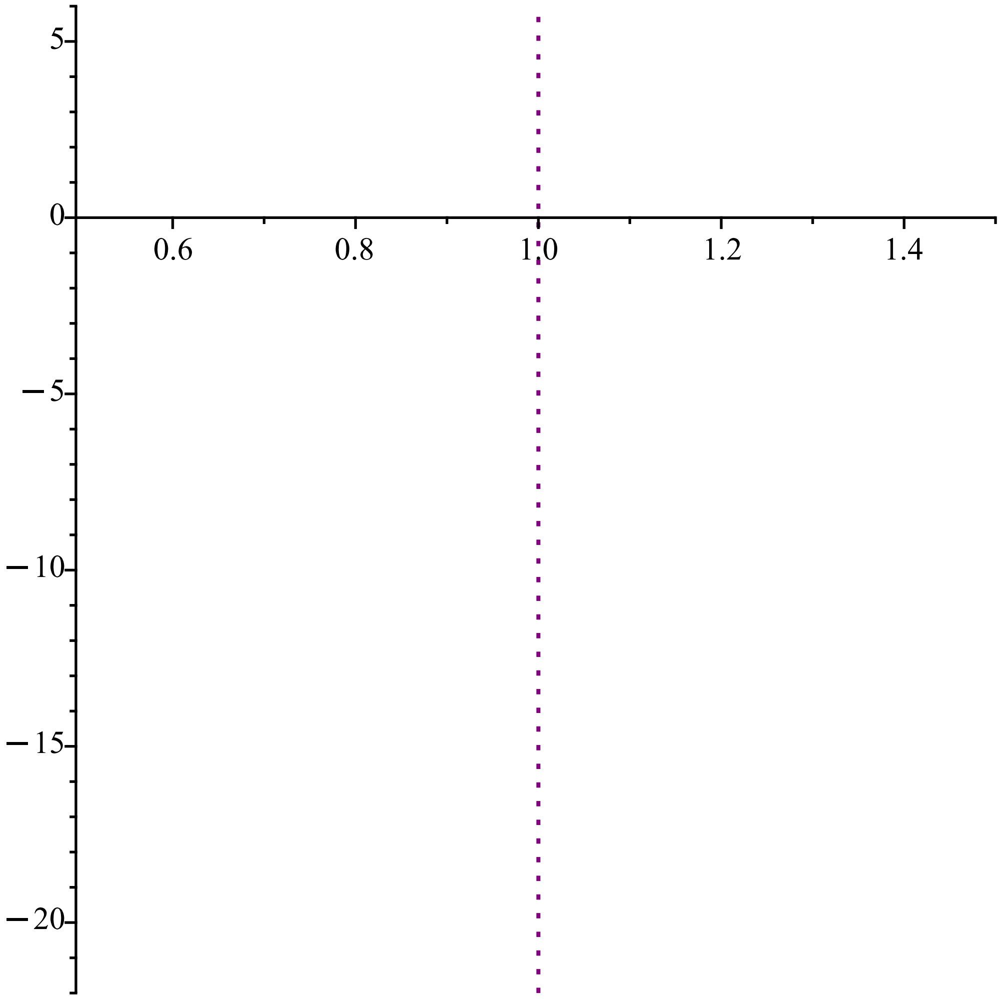
0
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
x
−1
−2
−3
−4
−5
6
F_Newton = -GM/r^2

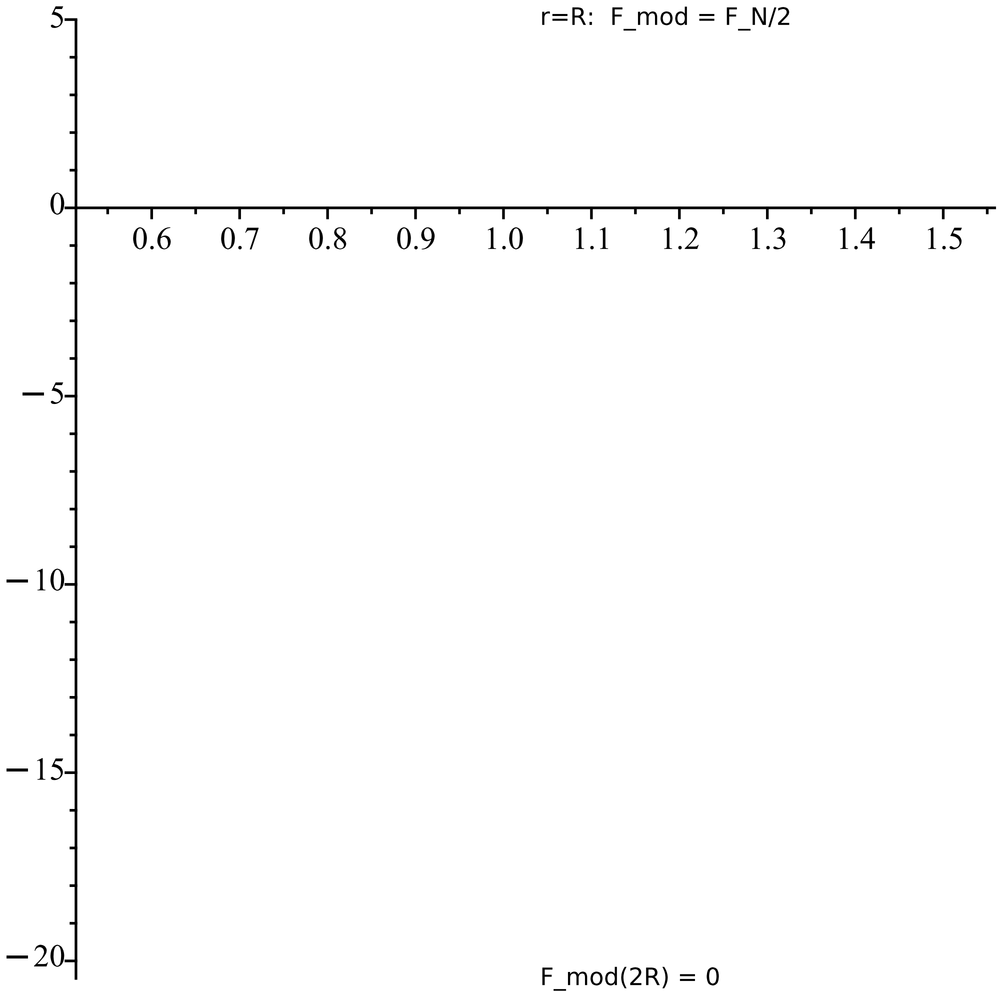


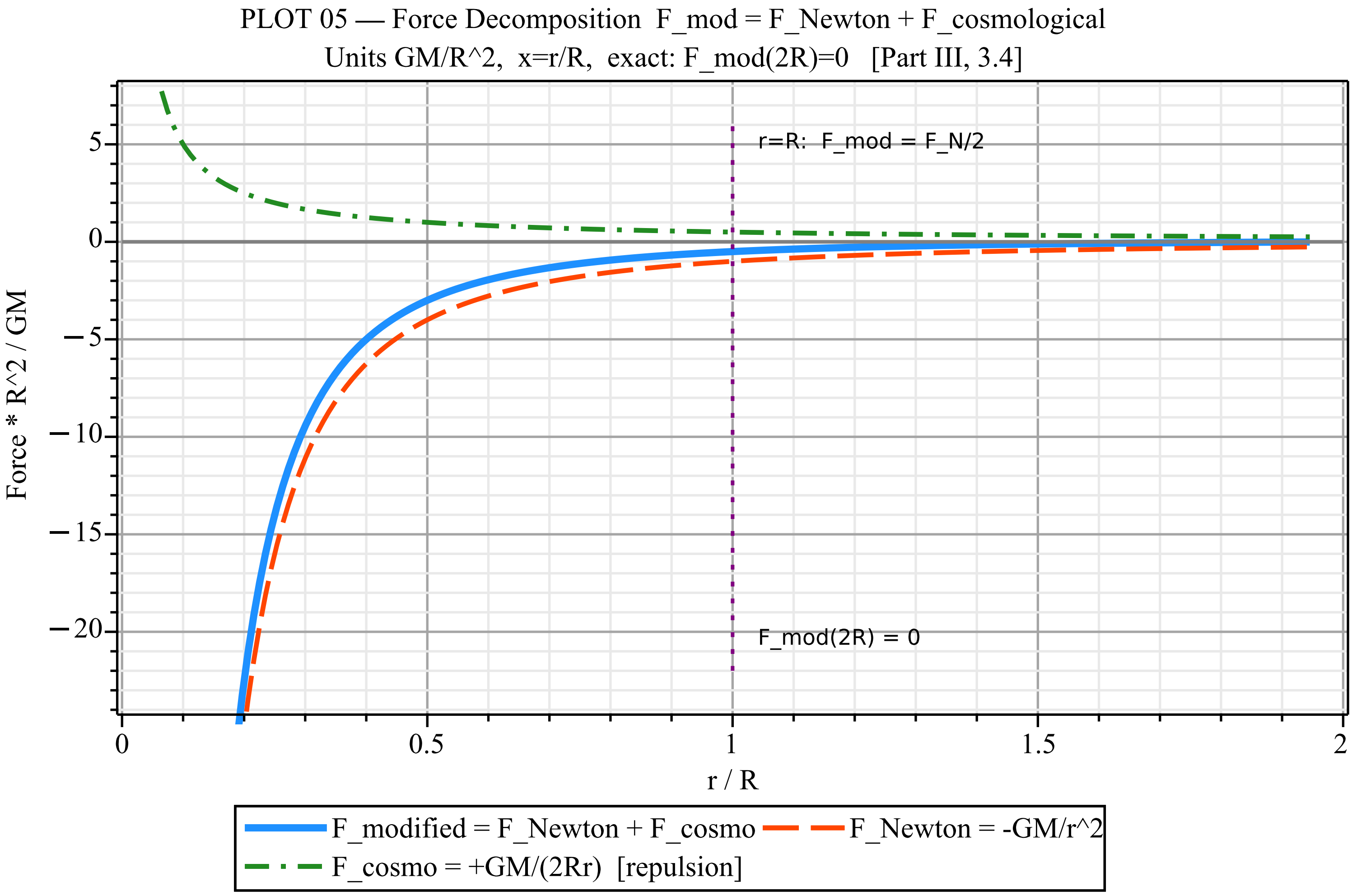
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6 1.8
x
F_cosmo = +GM/(2Rr) [repulsion]



1
0.5
0
−0.5
1
0.5 1 1.5 2
x

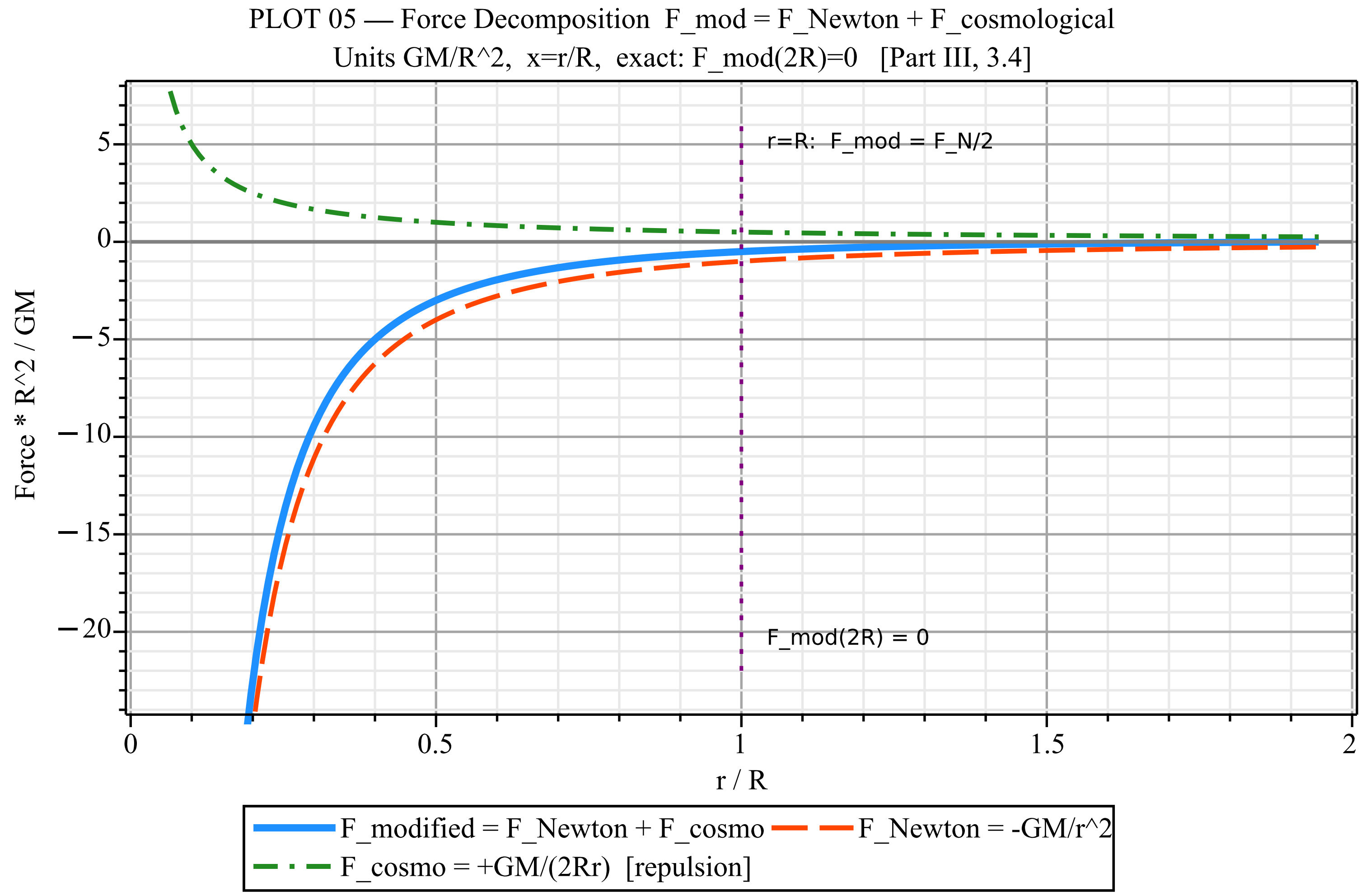

r=R: F_mod = F_N/2


F_mod(2R) = 0

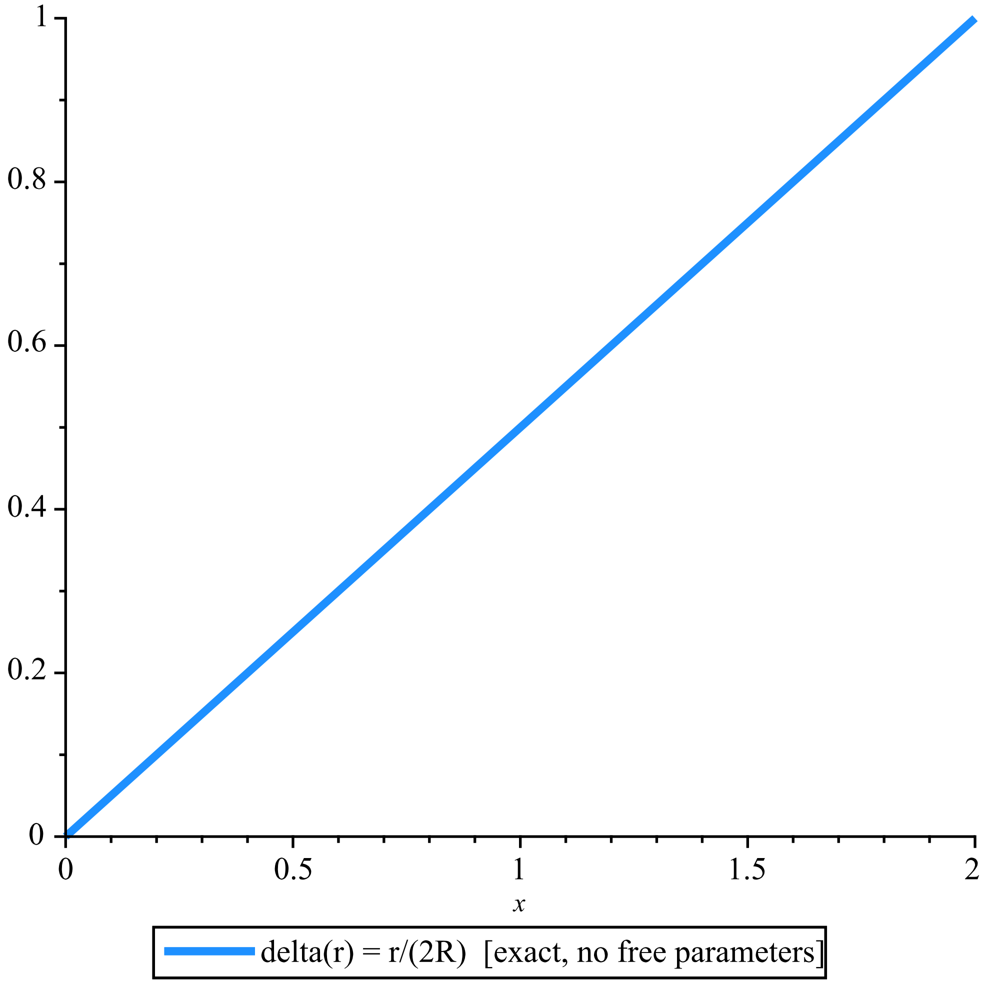


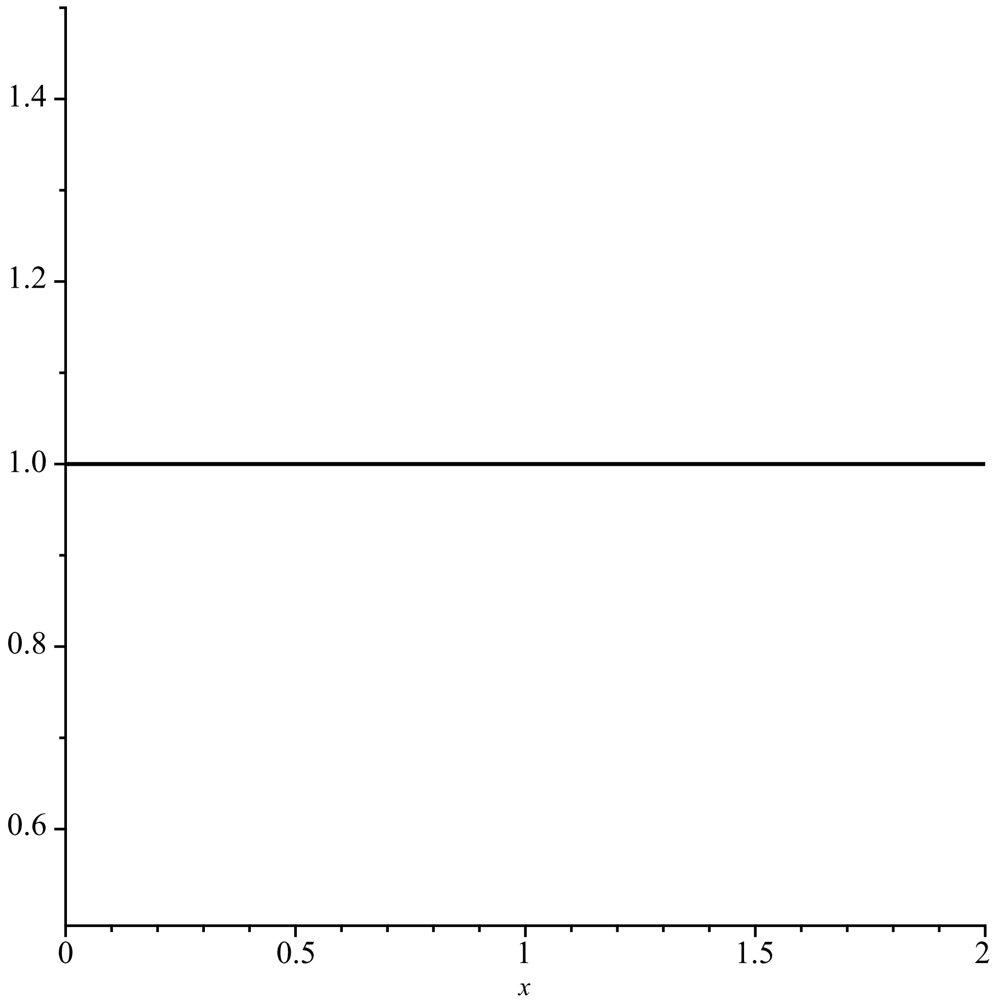
PLOT 05 — Force Decomposition F_mod = F_Newton + F_cosmological
Units GM/R^2, x=r/R, exact: F_mod(2R)=0 [Part III, 3.4]
r=R: F_mod = F_N/2
F_mod(2R) = 0
Force * R^2 / GM
5
0
−5
−10
−15
−20
0
0.5
1
1.5
2
r / R
F_modified = F_Newton + F_cosmo
F_Newton = -GM/r^2
F_cosmo = +GM/(2Rr) [repulsion]

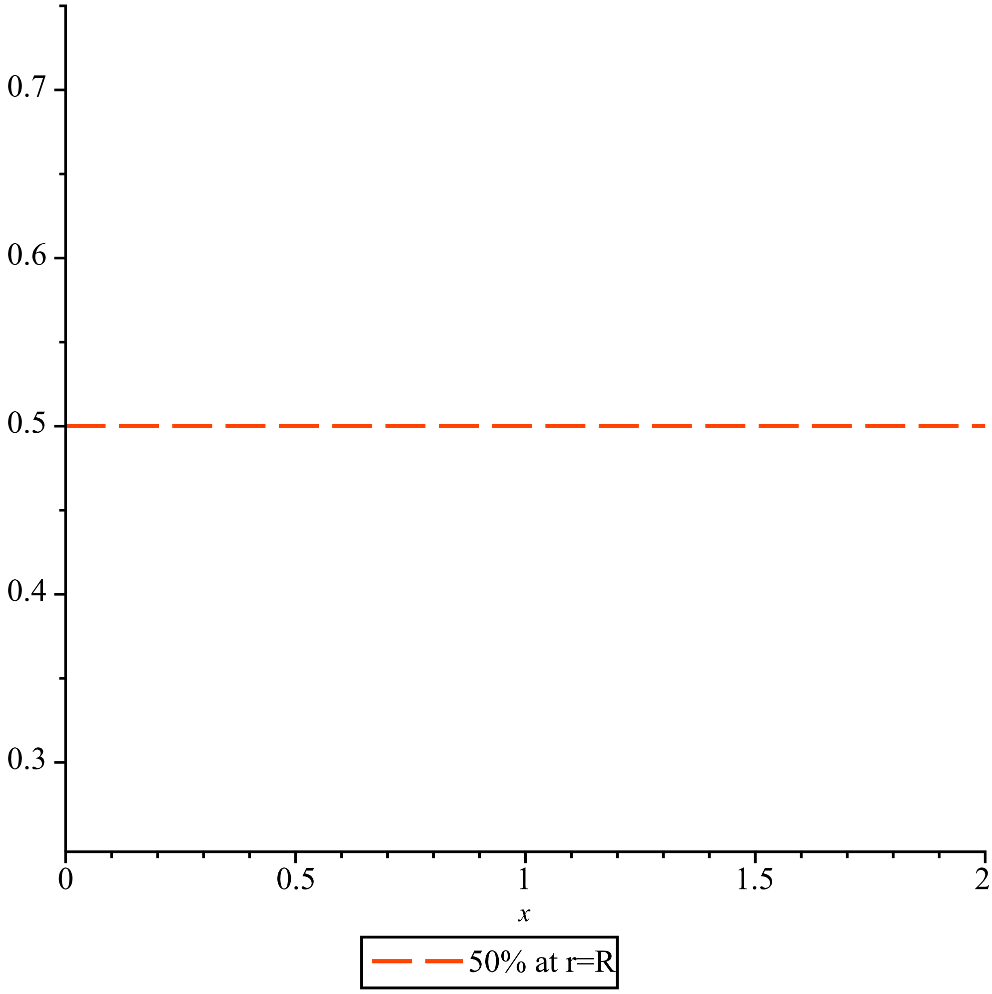
"PLOT 05 done."

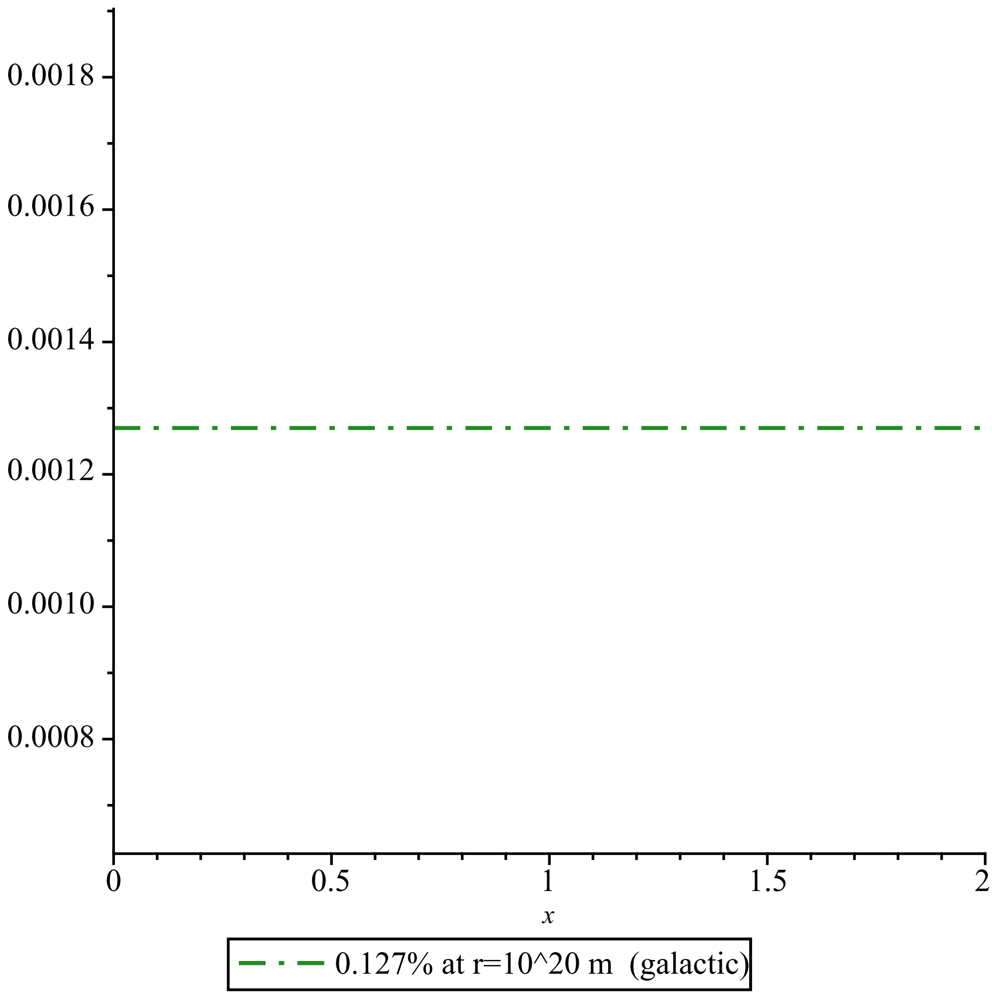


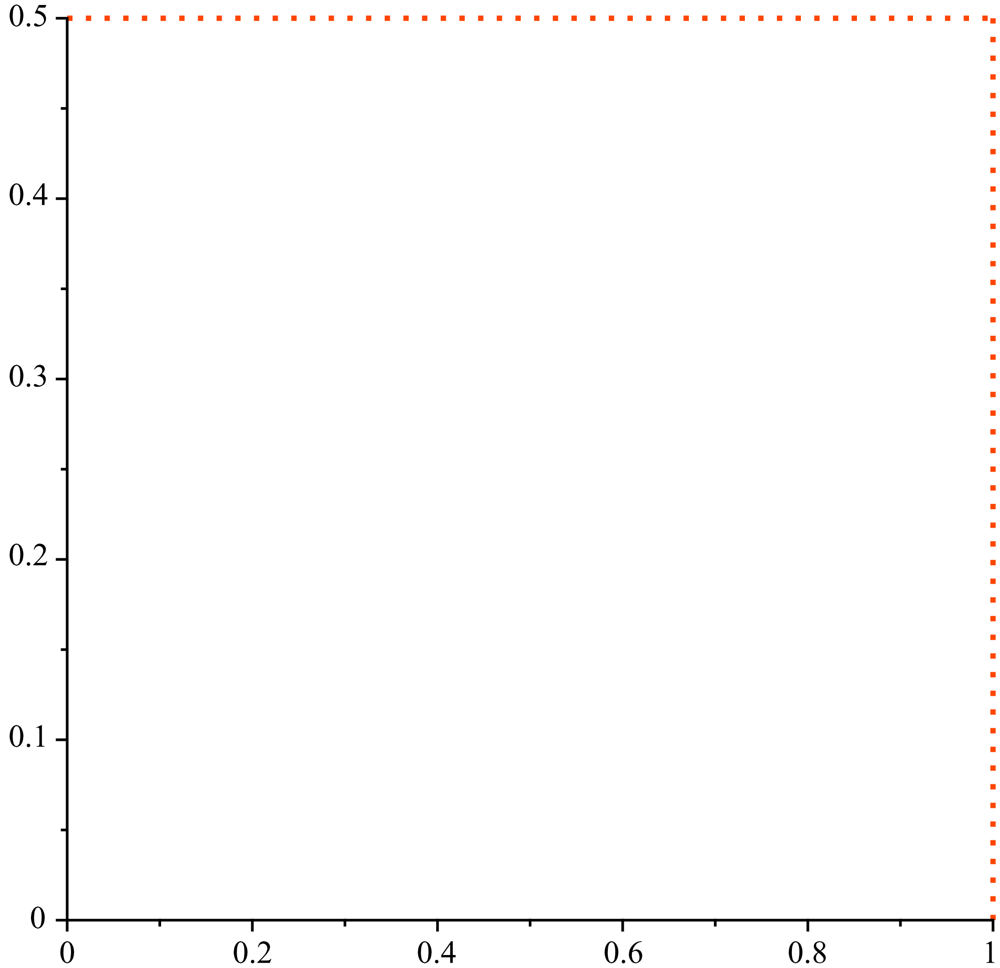

```
================================================================
 PLOT 06: Relative Deviation delta(r)=r/(2R)   [linear]
================================================================
```

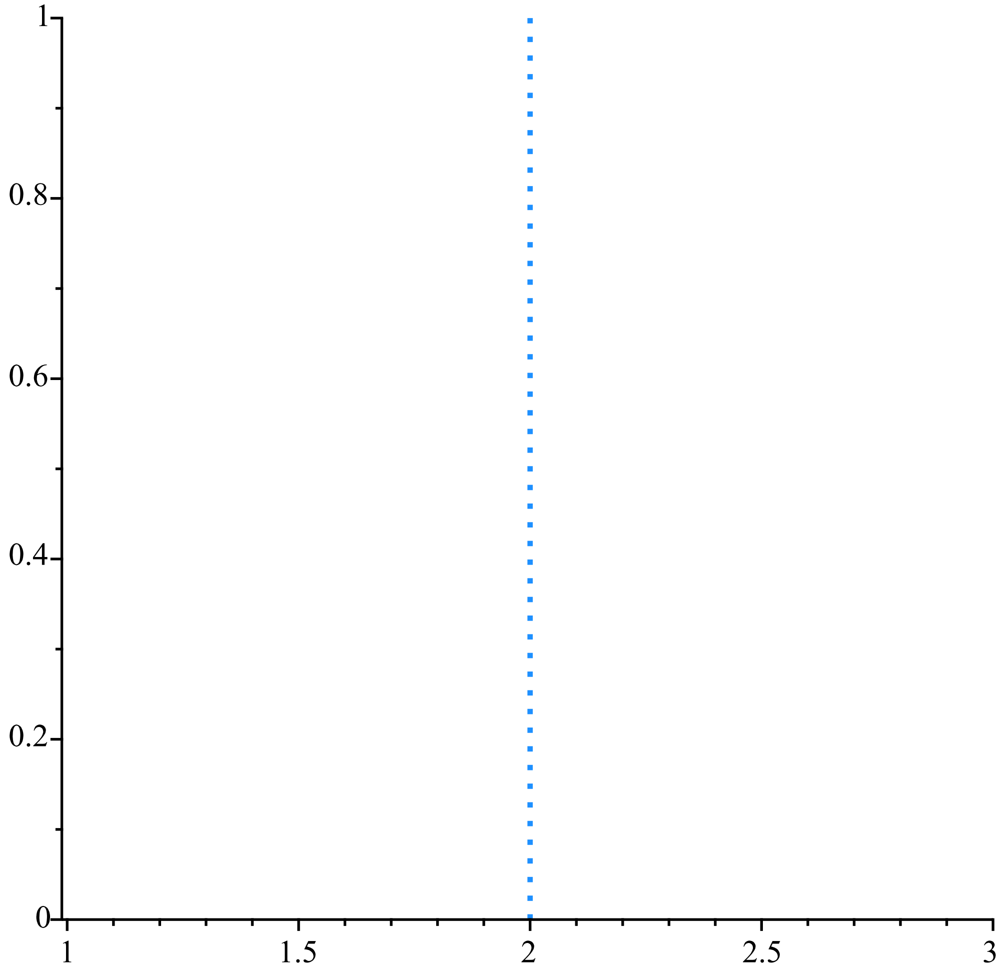

1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
x
delta(r) = r/(2R) [exact, no free parameters]

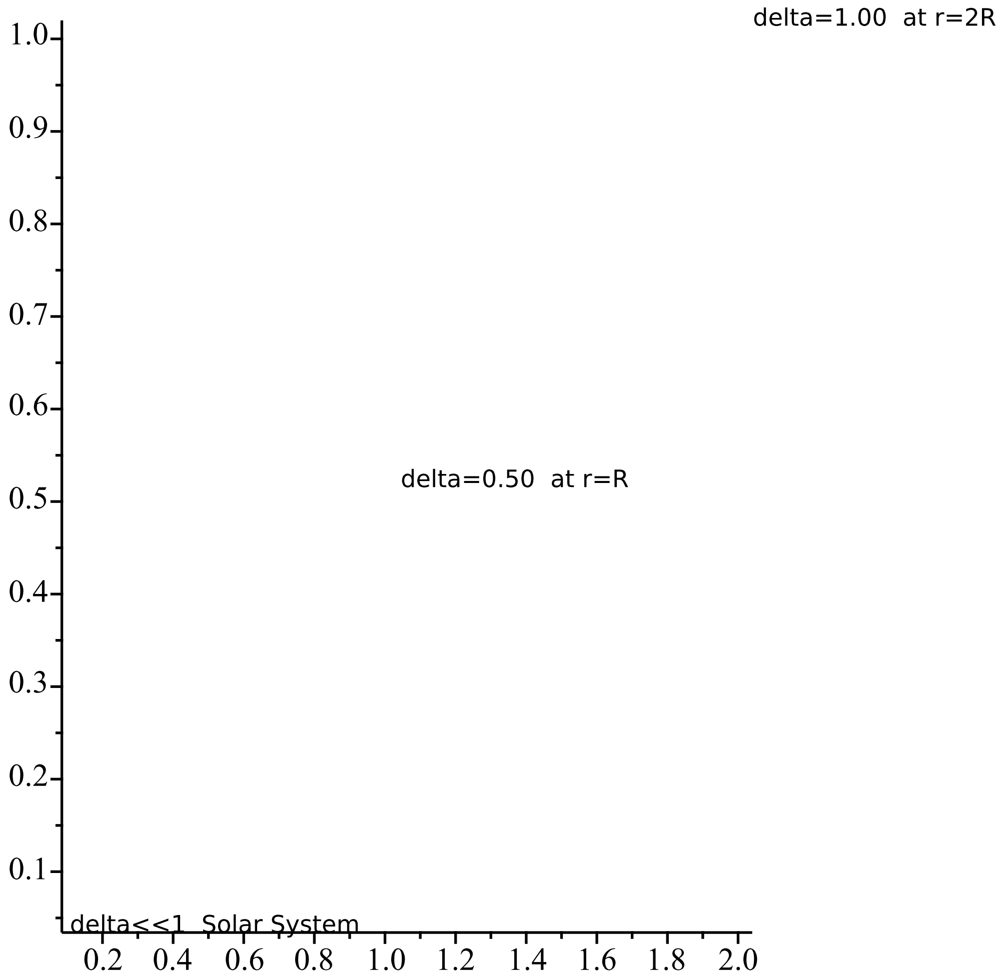

1.4
1.2
1.0
0.8
0.6
0
0.5
1
1.5
2
x


0.7
0.6
0.5
0.4
0.3
0
0.5
1
1.5
2
x
50% at r=R

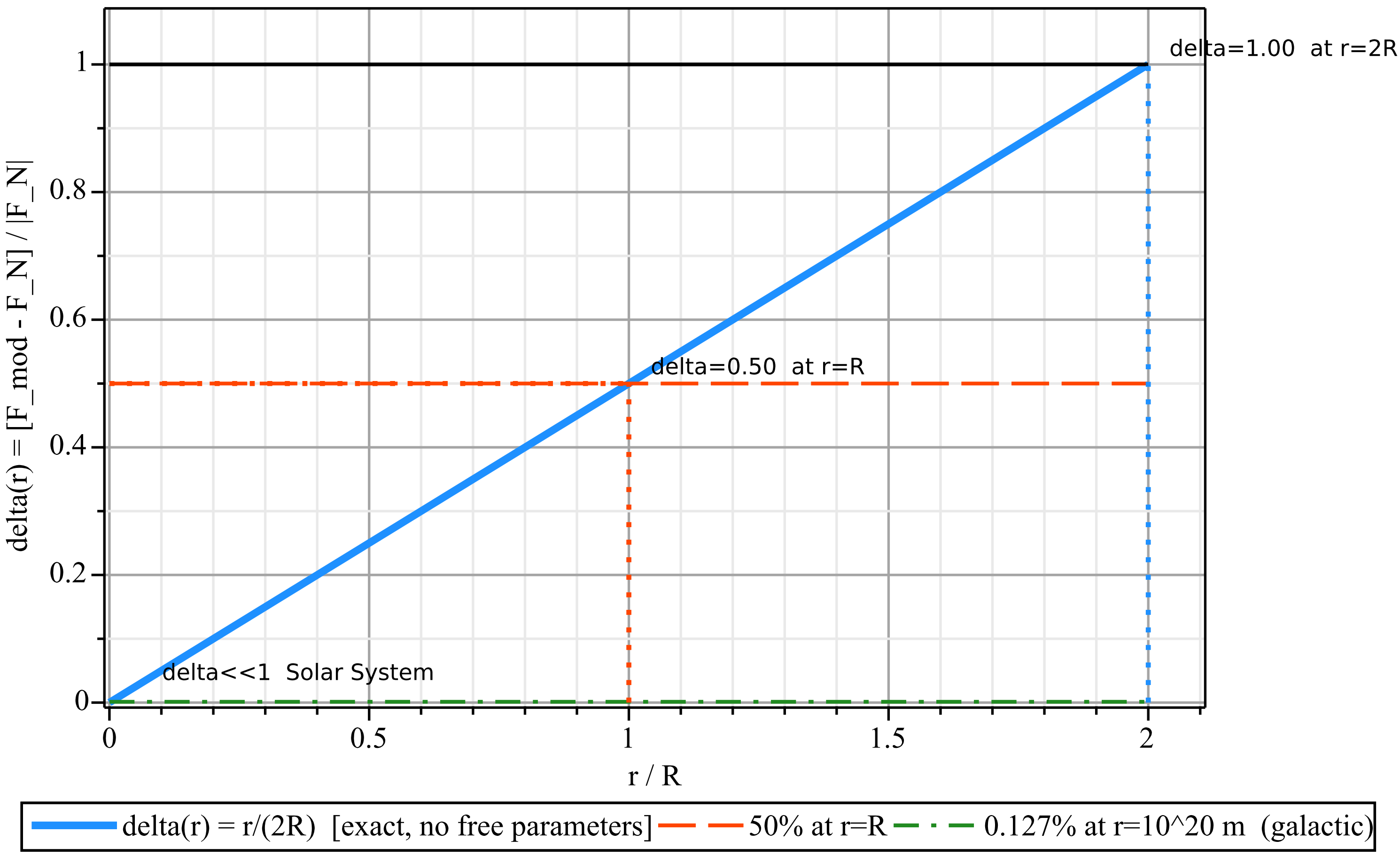

0.0018
0.0016
0.0014
0.0012
0.0010
0.0008
0
0.5
1
1.5
2
x
0.127% at r=10^20 m (galactic)


0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1

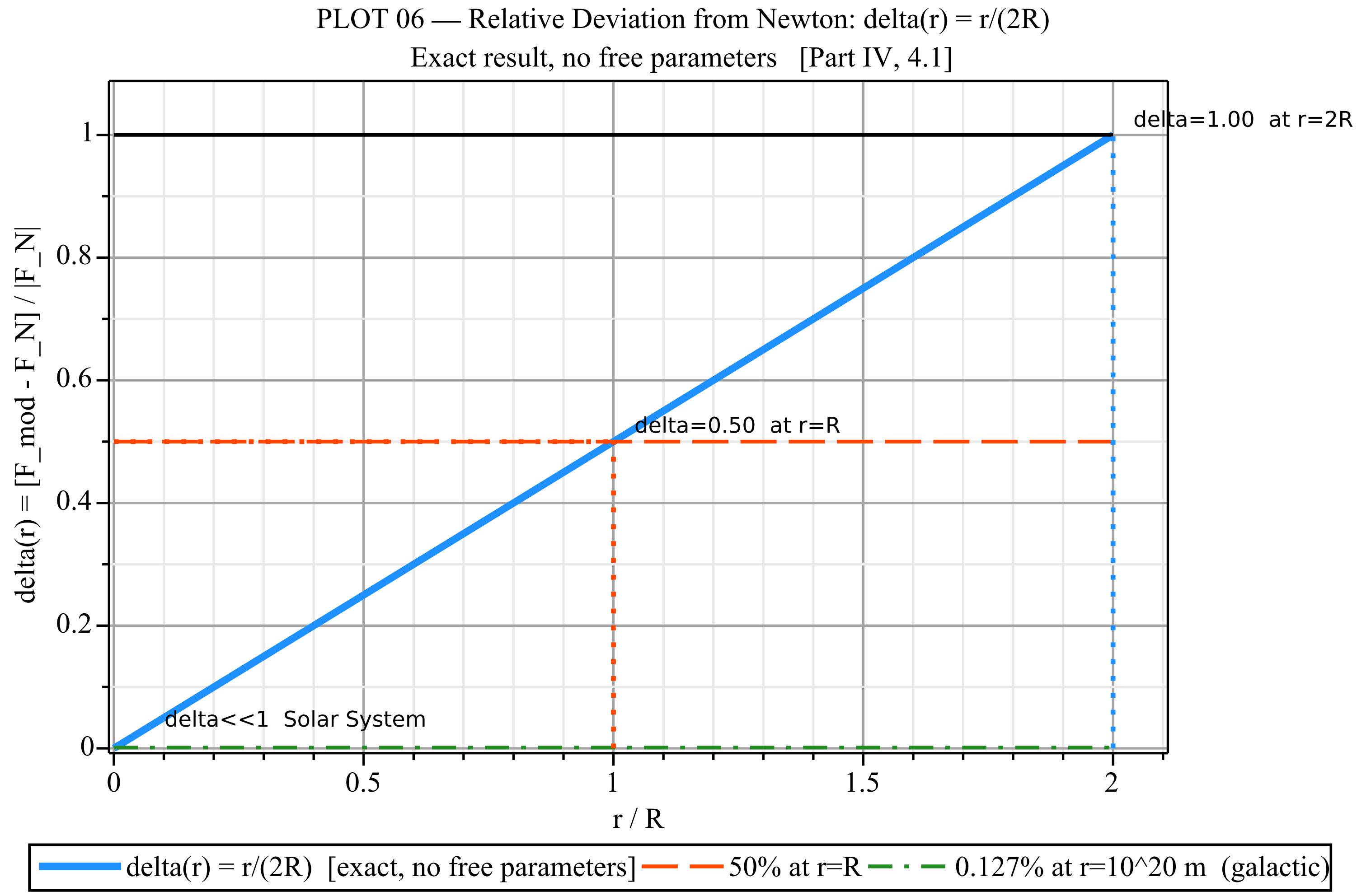

1
0.8
0.6
0.4
0.2
0
1
1.5
2
2.5
3



PLOT 06 — Relative Deviation from Newton: delta(r) = r/(2R)
Exact result, no free parameters   [Part IV, 4.1]



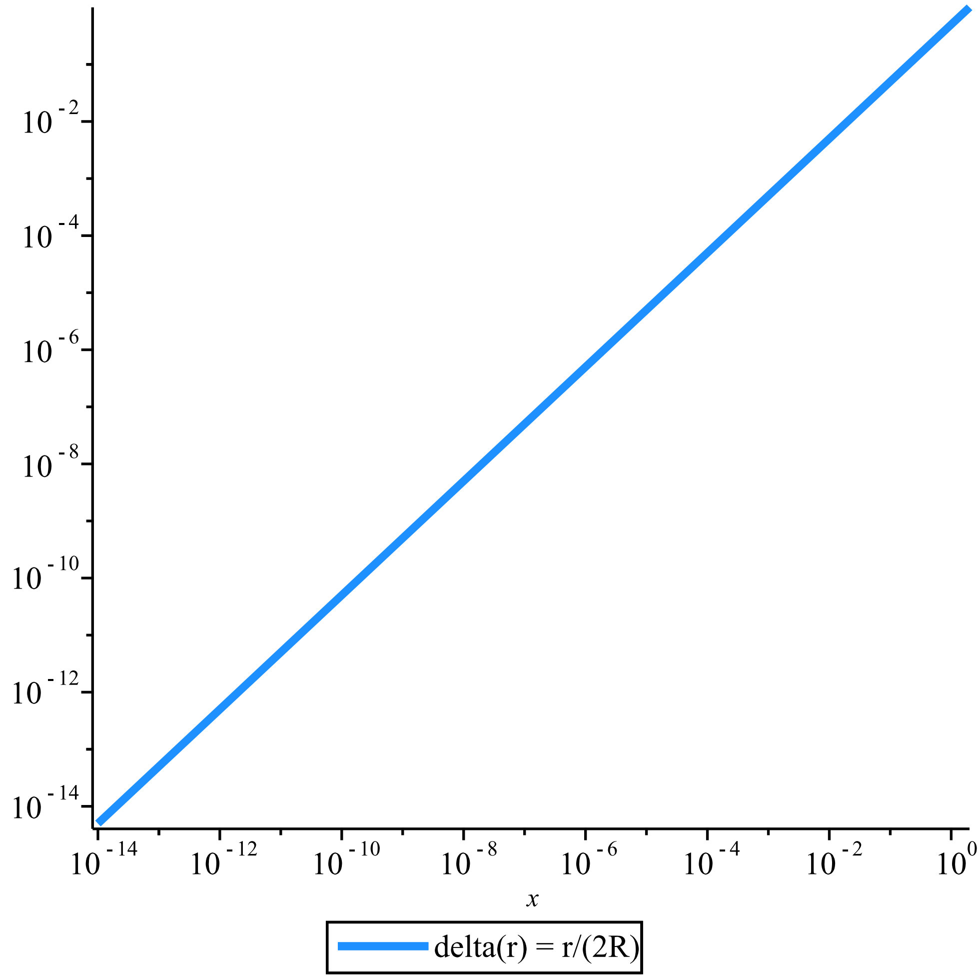
"PLOT 06 done."

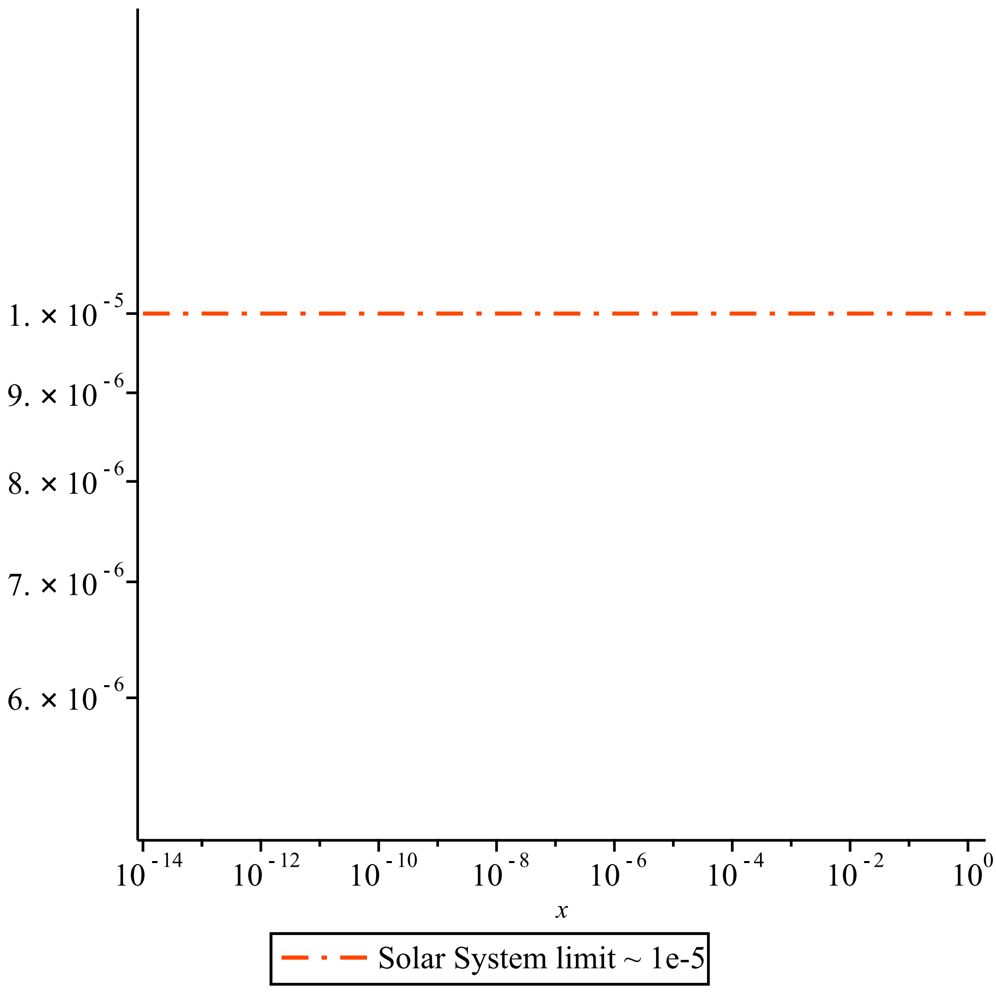


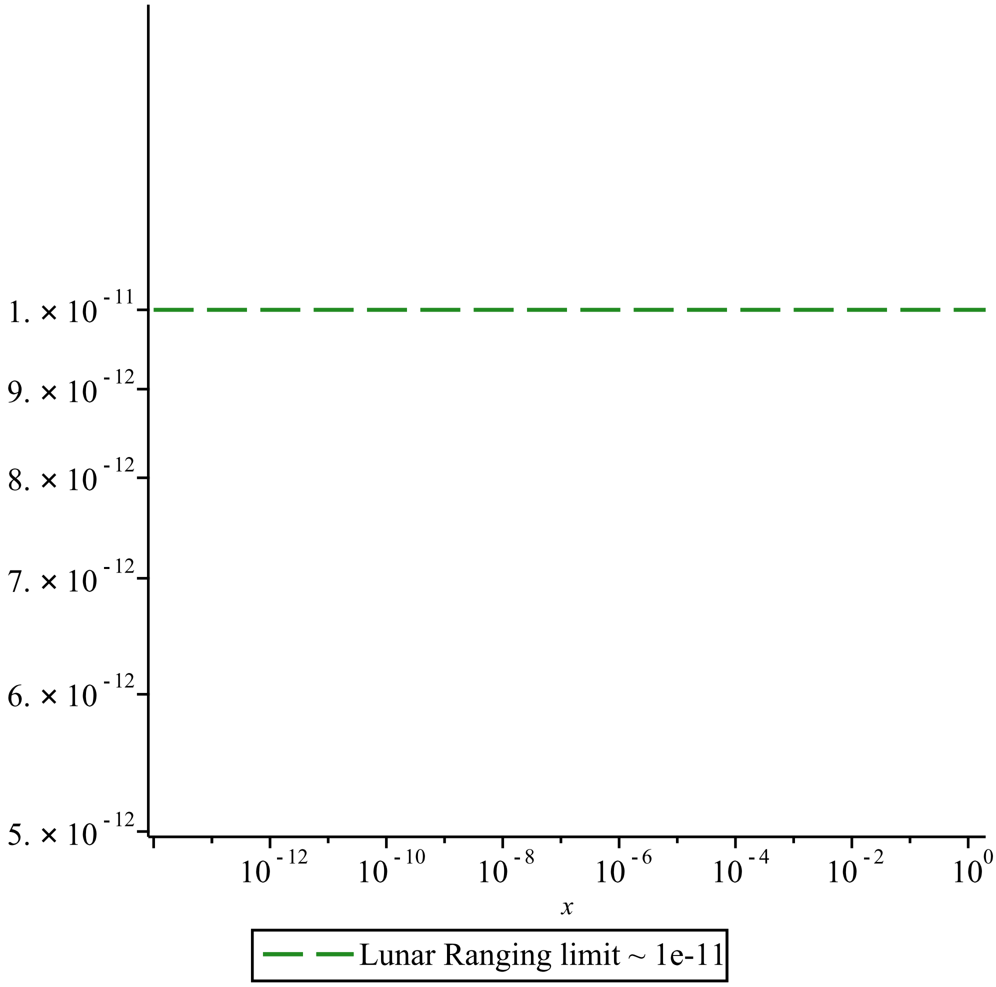
==

PLOT 07: delta(r) log-log full astronomical range

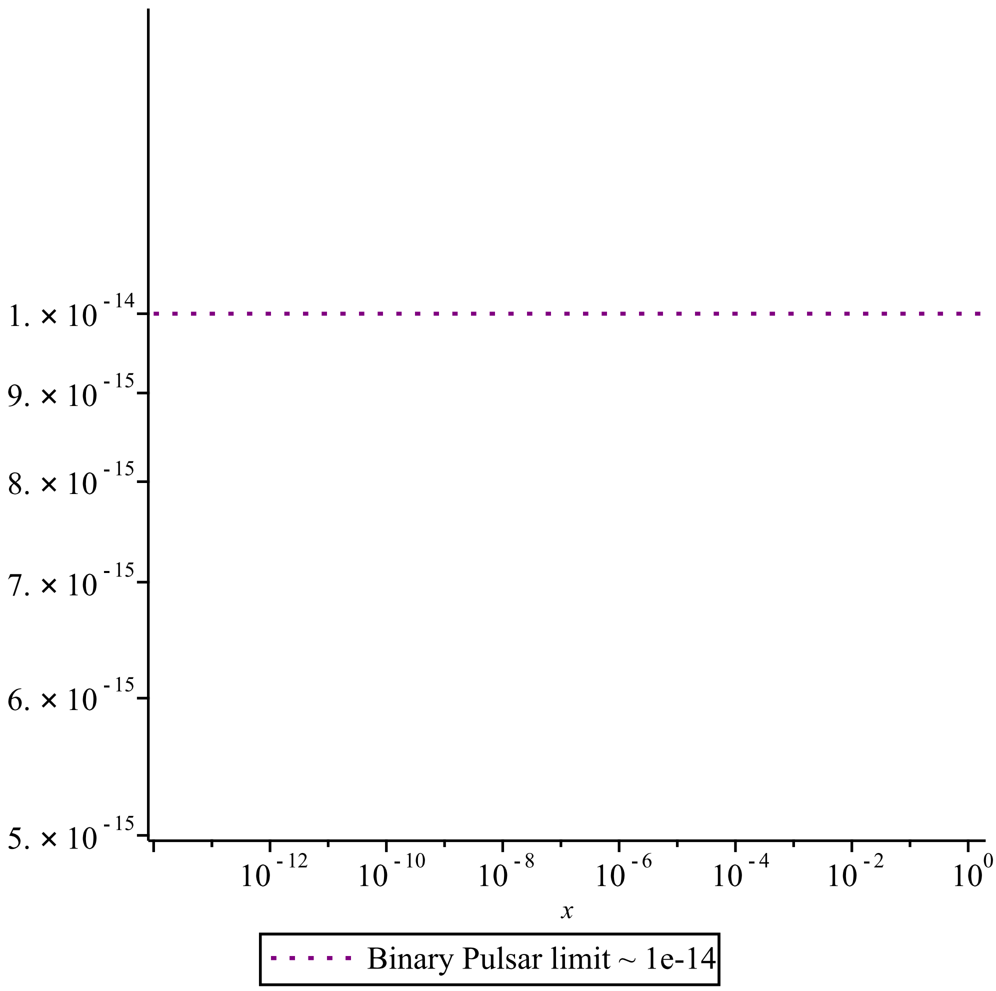
==


10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
10^-14 10^-12 10^-10 10^-8 10^-6 10^-4 10^-2 10^0
x
delta(r) = r/(2R)

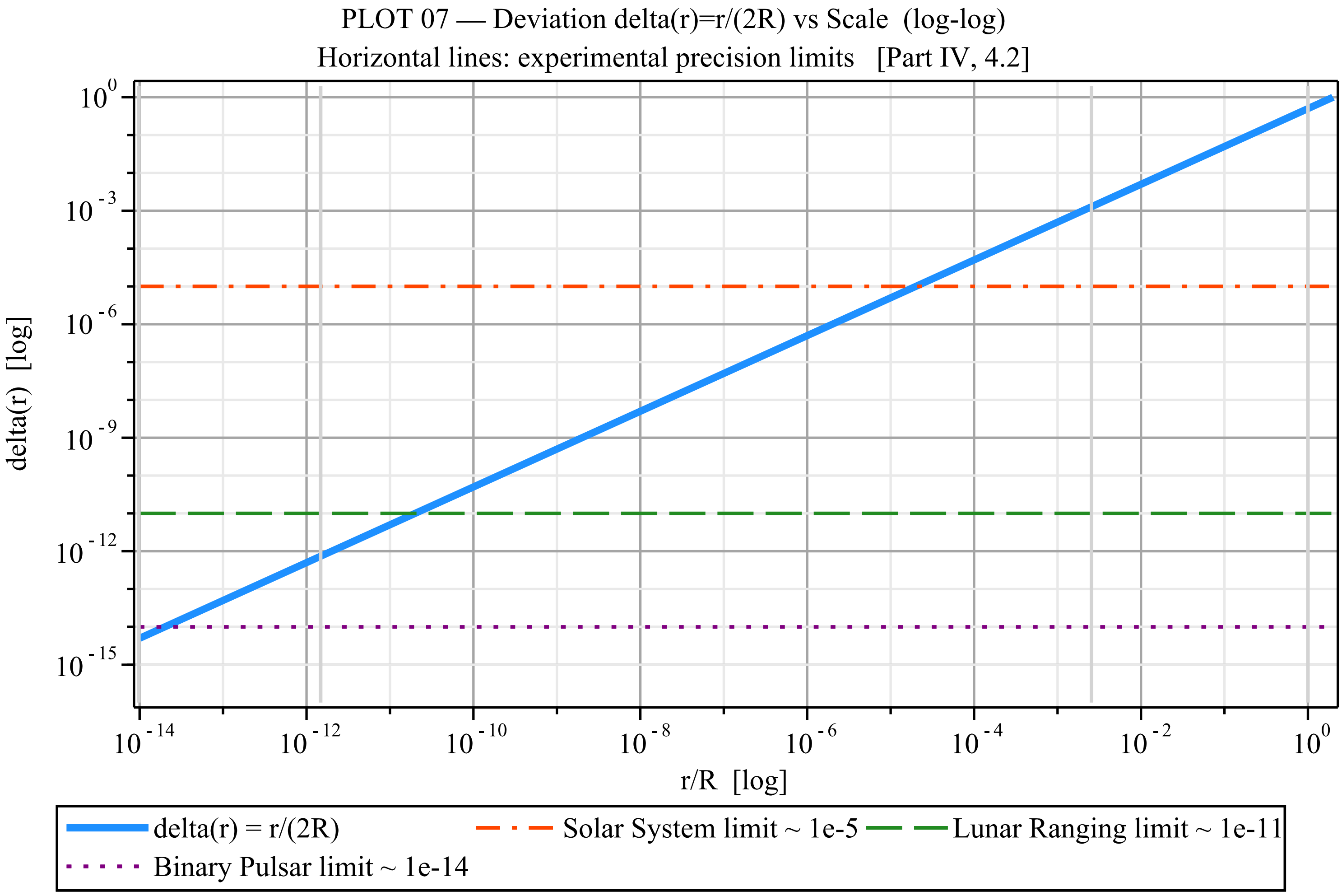

1. × 10^-5
9. × 10^-6
8. × 10^-6
7. × 10^-6
6. × 10^-6
10^-14 10^-12 10^-10 10^-8 10^-6 10^-4 10^-2 10^0
x
Solar System limit ~ 1e-5



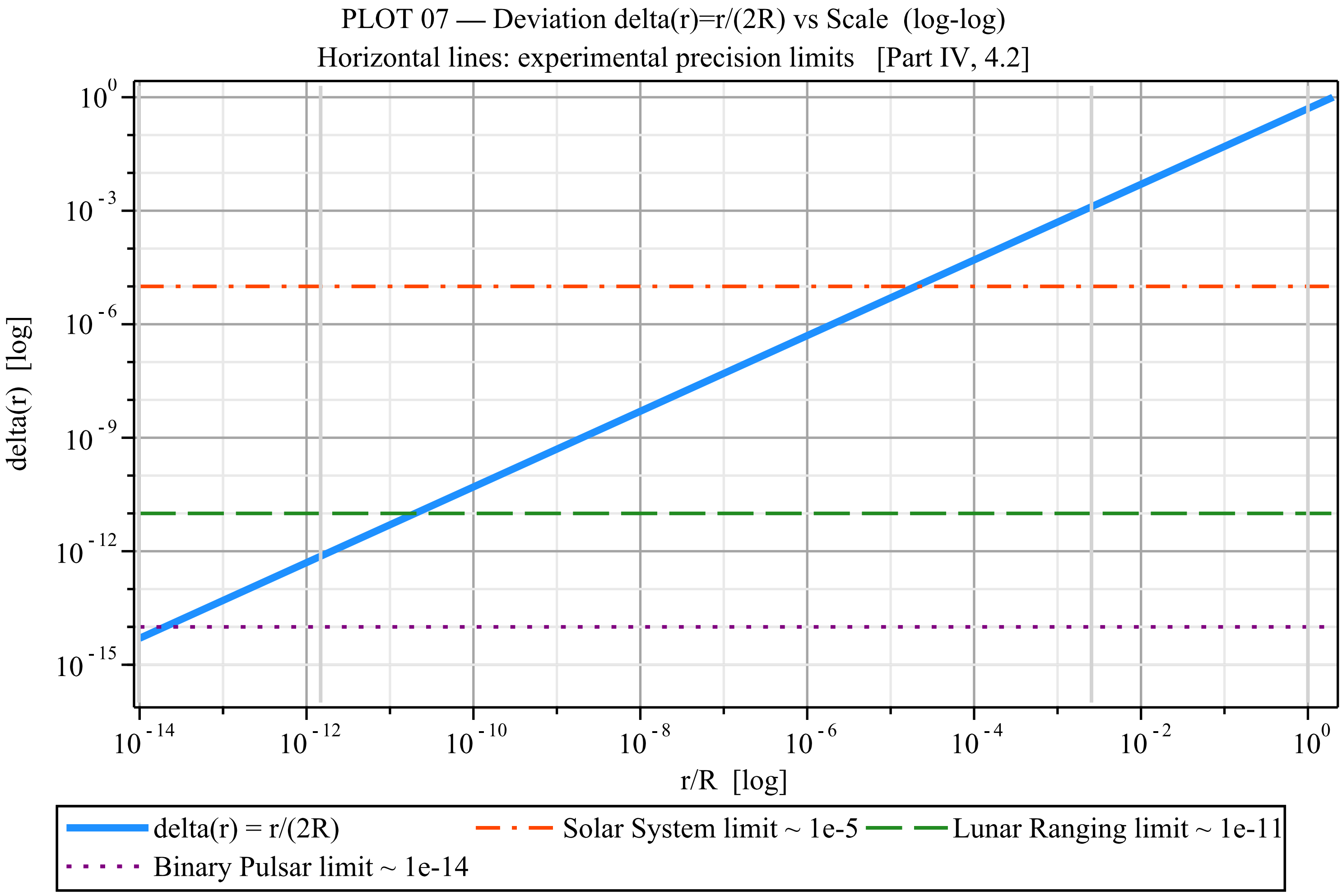
1. × 10^-11
9. × 10^-12
8. × 10^-12
7. × 10^-12
6. × 10^-12
5. × 10^-12
10^-12
10^-10
10^-8
10^-6
10^-4
10^-2
10^0
x
Lunar Ranging limit ~ 1e-11



1. × 10^-14
9. × 10^-15
8. × 10^-15
7. × 10^-15
6. × 10^-15
5. × 10^-15
10^-12
10^-10
10^-8
10^-6
10^-4
10^-2
10^0
x
Binary Pulsar limit ~ 1e-14


PLOT 07 — Deviation delta(r)=r/(2R) vs Scale (log-log)
Horizontal lines: experimental precision limits [Part IV, 4.2]
delta(r) [log]
r/R [log]
delta(r) = r/(2R)
Solar System limit ~ 1e-5
Lunar Ranging limit ~ 1e-11
Binary Pulsar limit ~ 1e-14

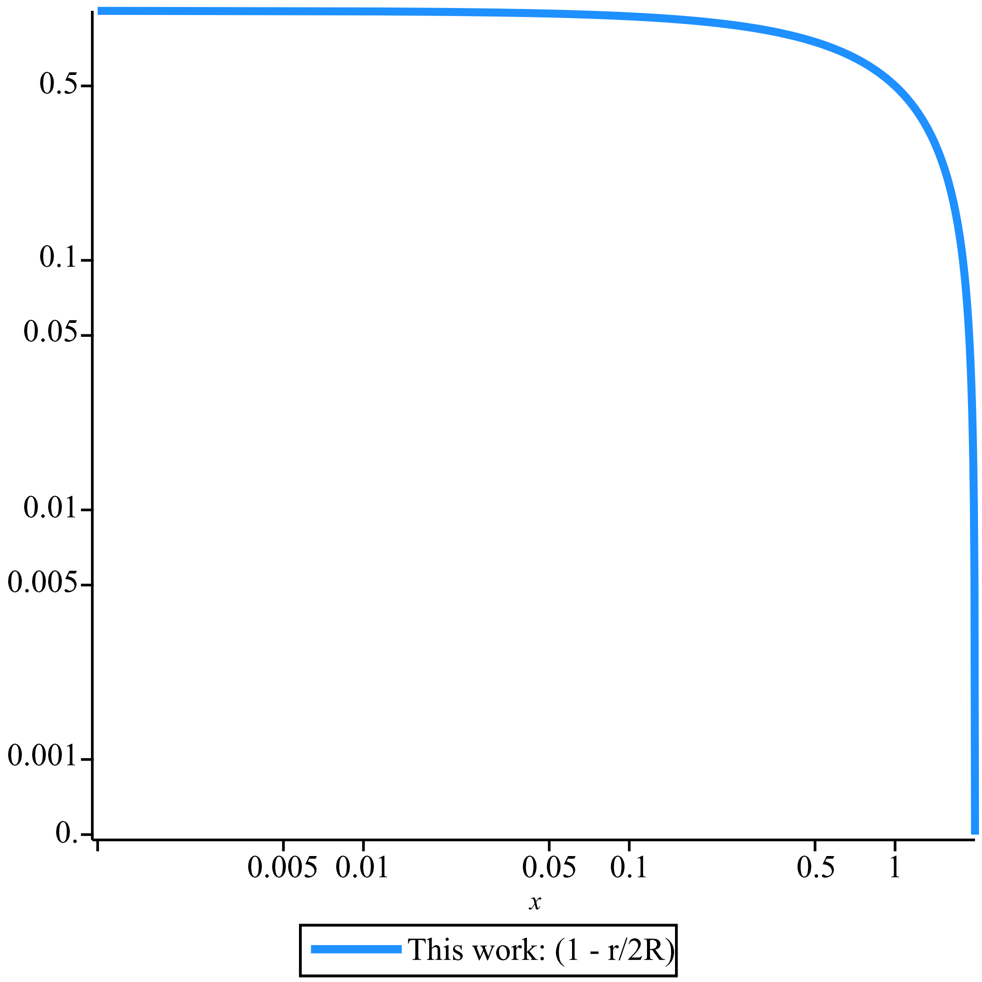
"PLOT 07 done."



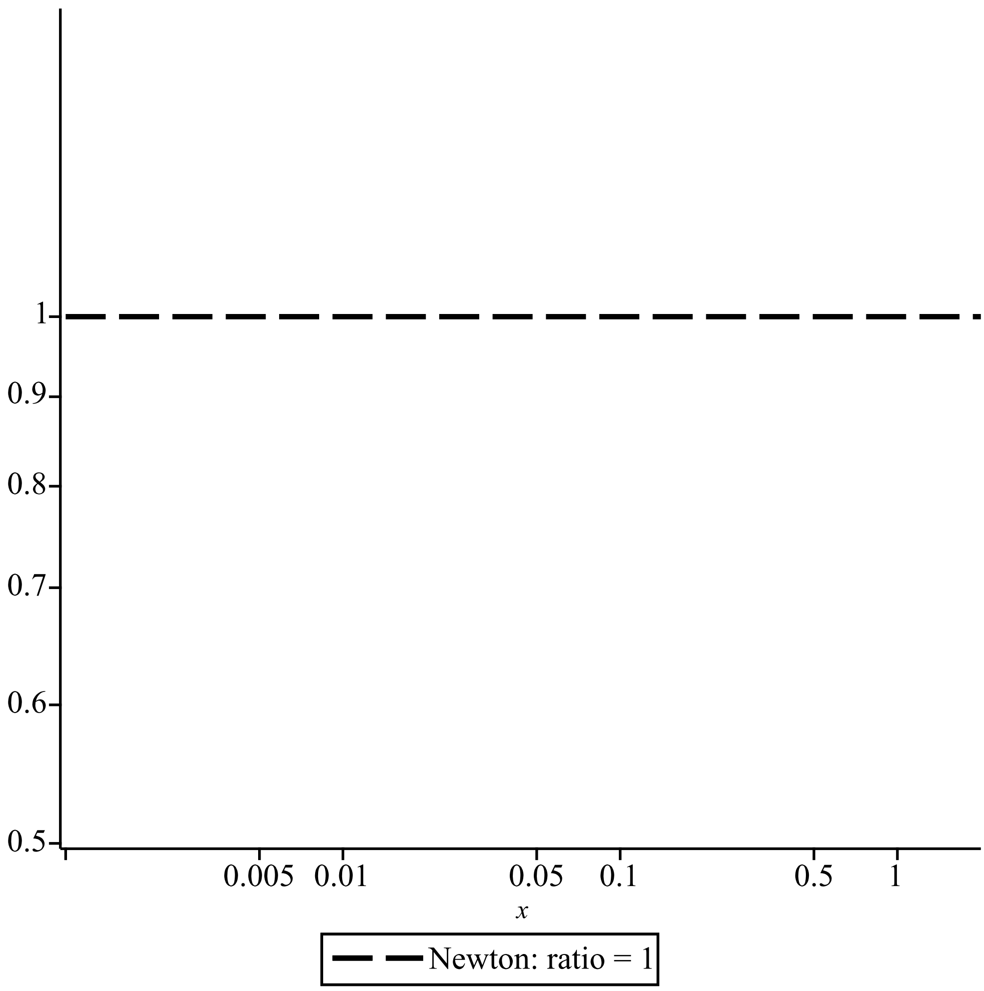
===

PLOT 08: Modified Gravity vs MOND vs Yukawa

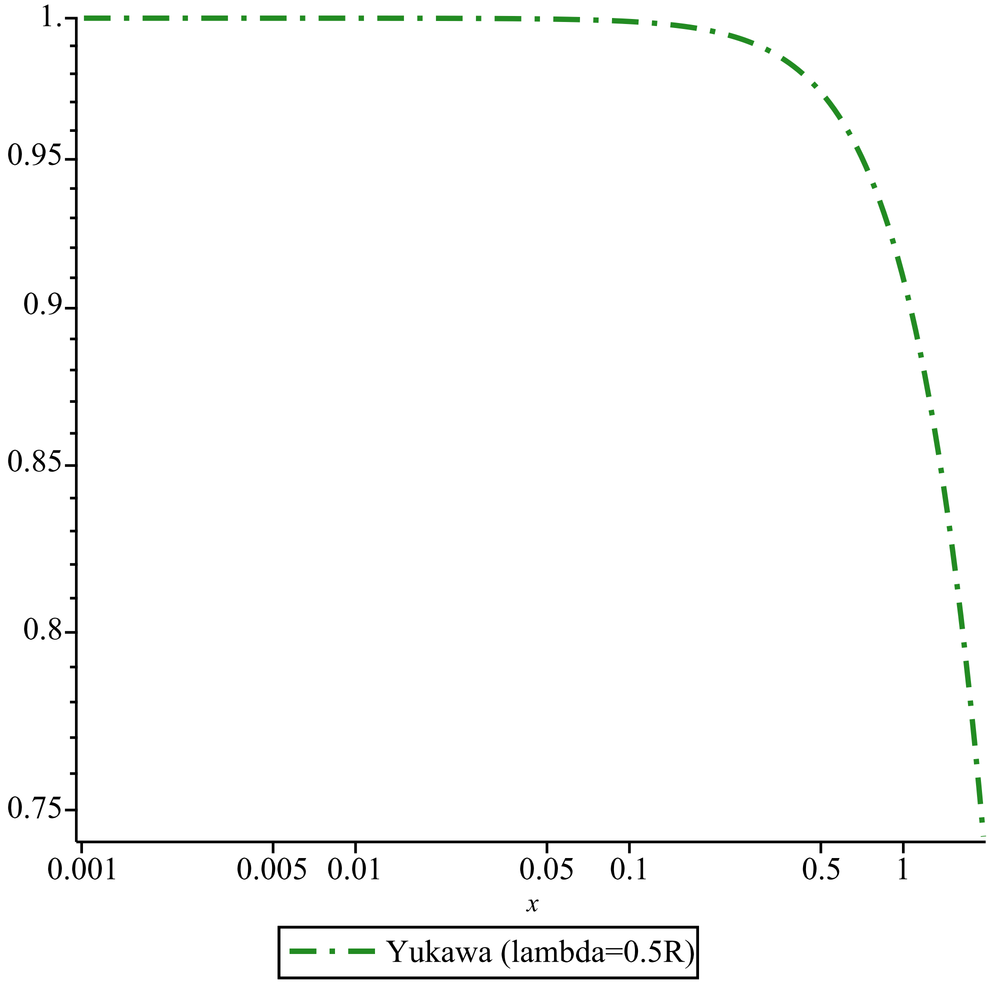
===

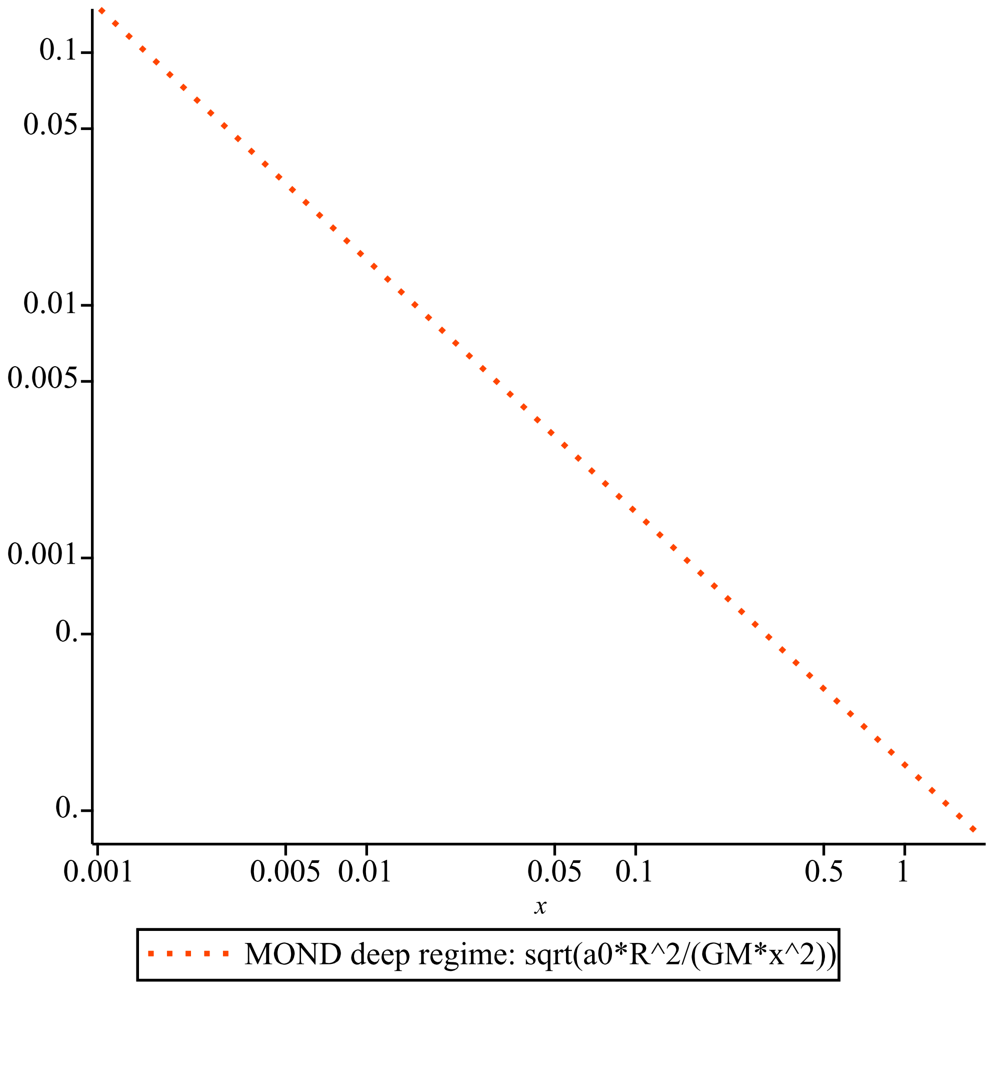


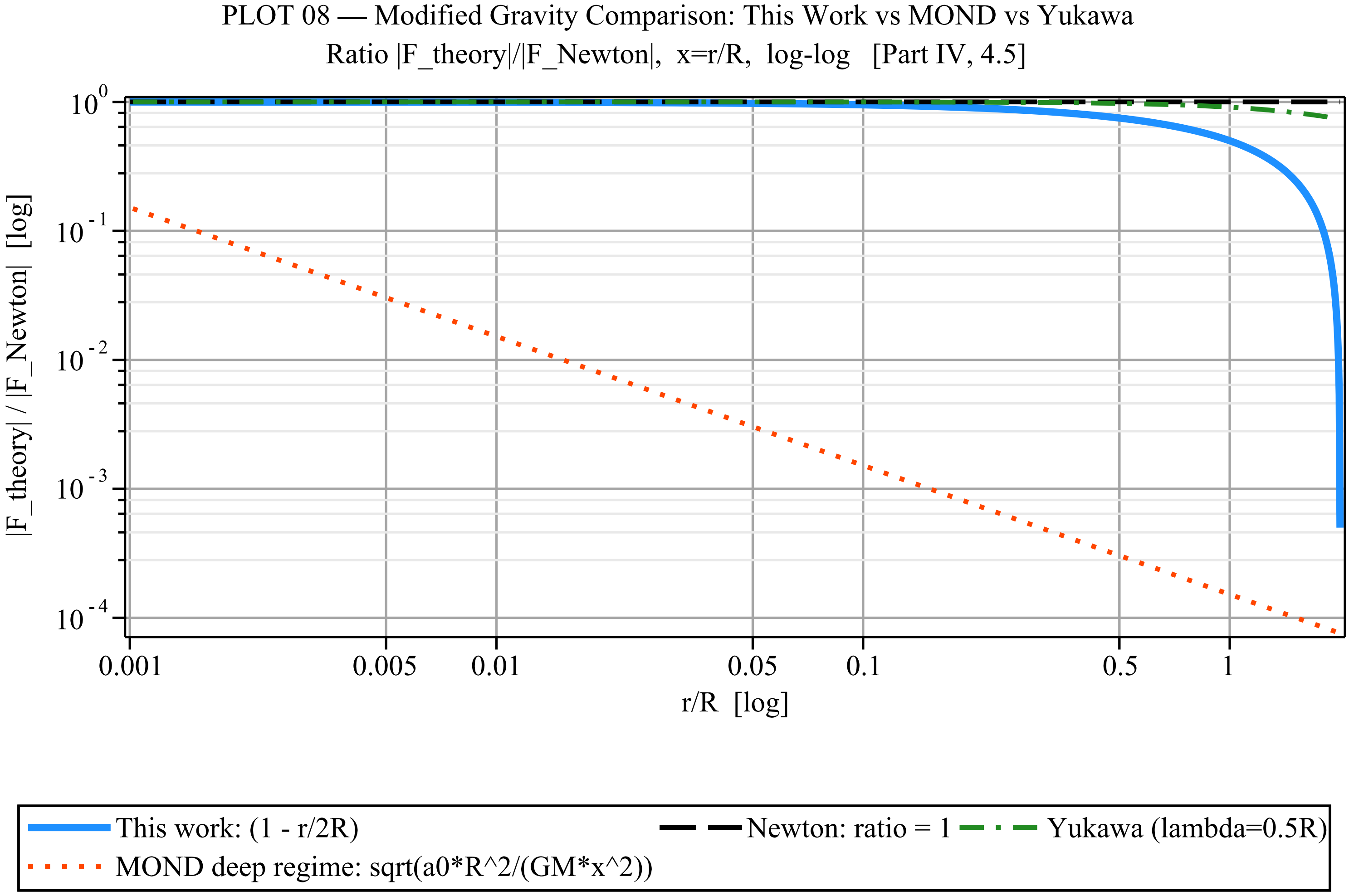
0.5
0.1
0.05
0.01
0.005
0.001
0.
0.005 0.01 0.05 0.1 0.5 1
x
This work: (1 - r/2R)



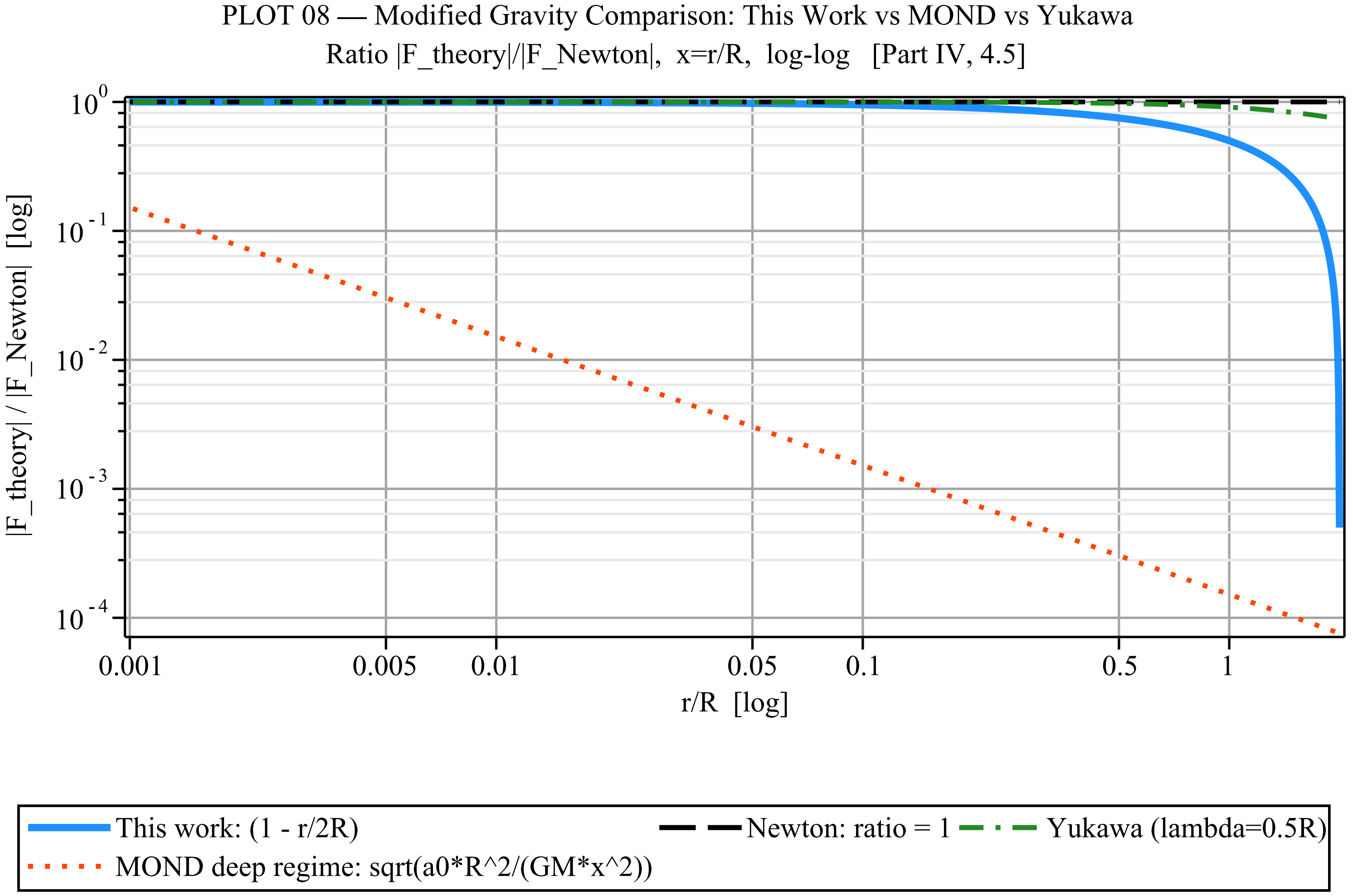
1
0.9
0.8
0.7
0.6
0.5
0.005 0.01 0.05 0.1 0.5 1
x
Newton: ratio = 1


1.
0.95
0.9
0.85
0.8
0.75
0.001
0.005
0.01
0.05
0.1
0.5
1
x
Yukawa (lambda=0.5R)

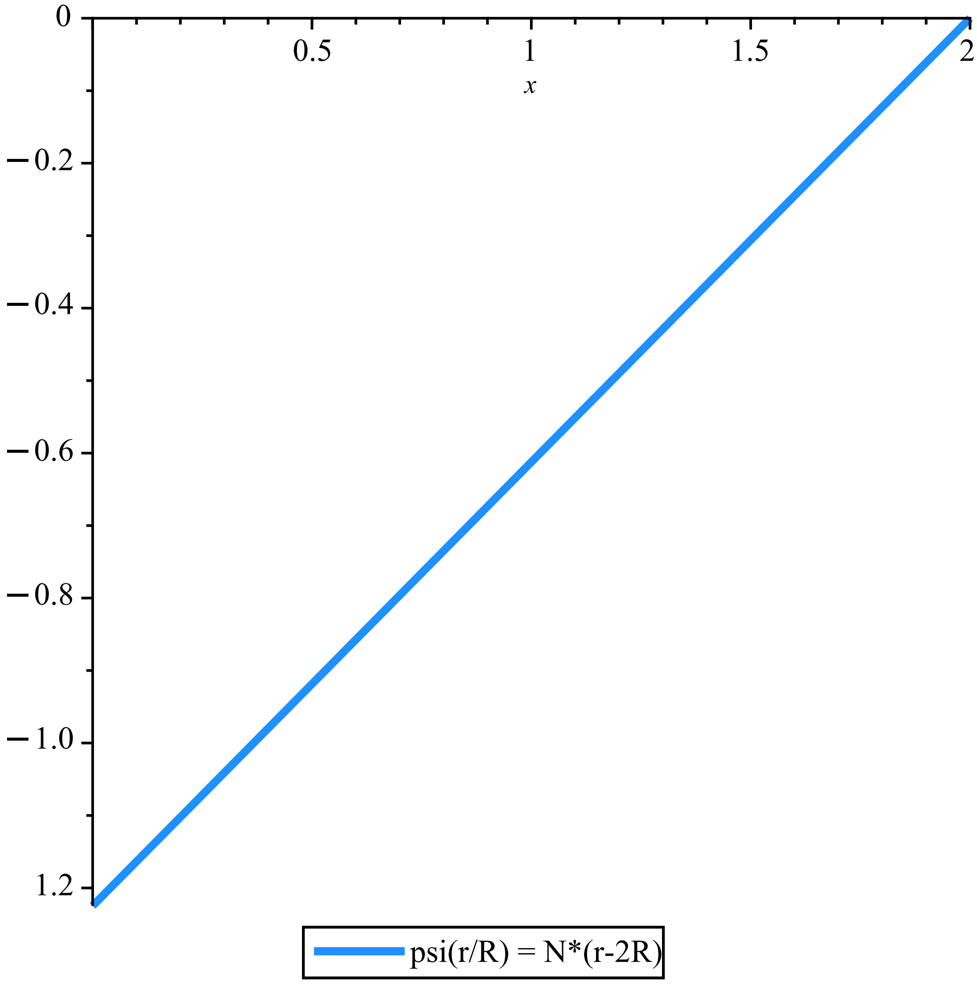

0.1
0.05
0.01
0.005
0.001
0.
0.
0.001
0.005
0.01
0.05
0.1
0.5
1
x
MOND deep regime: sqrt(a0*R^2/(GM*x^2))


PLOT 08 — Modified Gravity Comparison: This Work vs MOND vs Yukawa
Ratio |F_theory|/|F_Newton|, x=r/R, log-log [Part IV, 4.5]
|F_theory| / |F_Newton| [log]
10^0
10^-1
10^-2
10^-3
10^-4
0.001
0.005
0.01
0.05
0.1
0.5
1
r/R [log]
This work: (1 - r/2R)
Newton: ratio = 1
Yukawa (lambda=0.5R)
MOND deep regime: sqrt(a0*R^2/(GM*x^2))

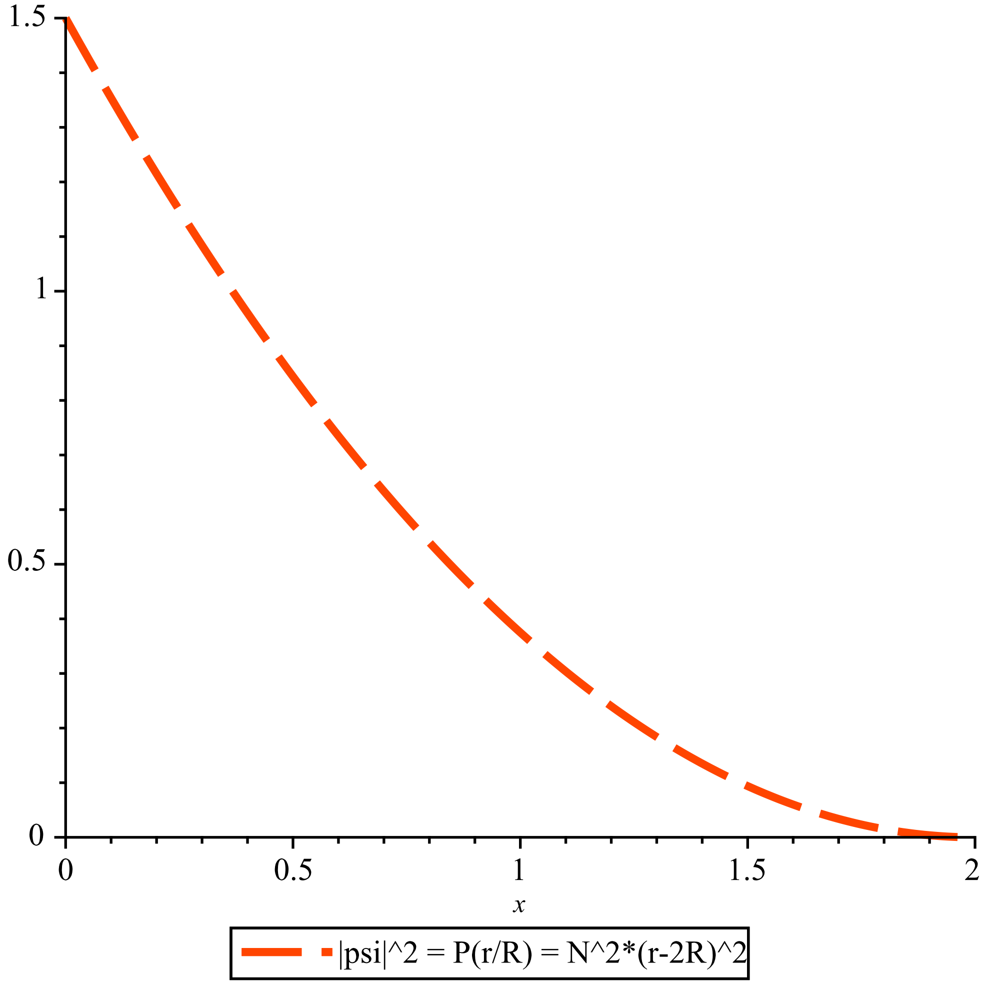
"PLOT 08 done."



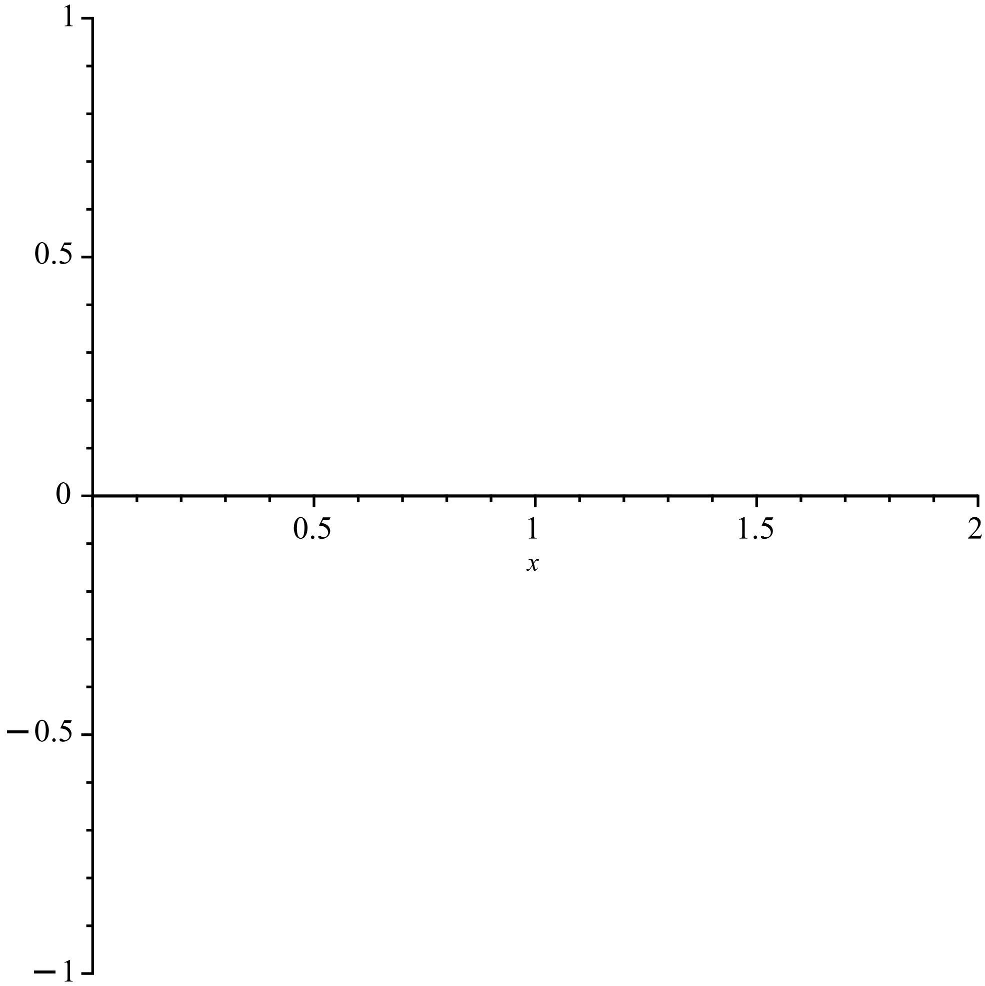

```
================================================================
 PLOT 09: Wave Function psi(r) and Probability Density
================================================================
```

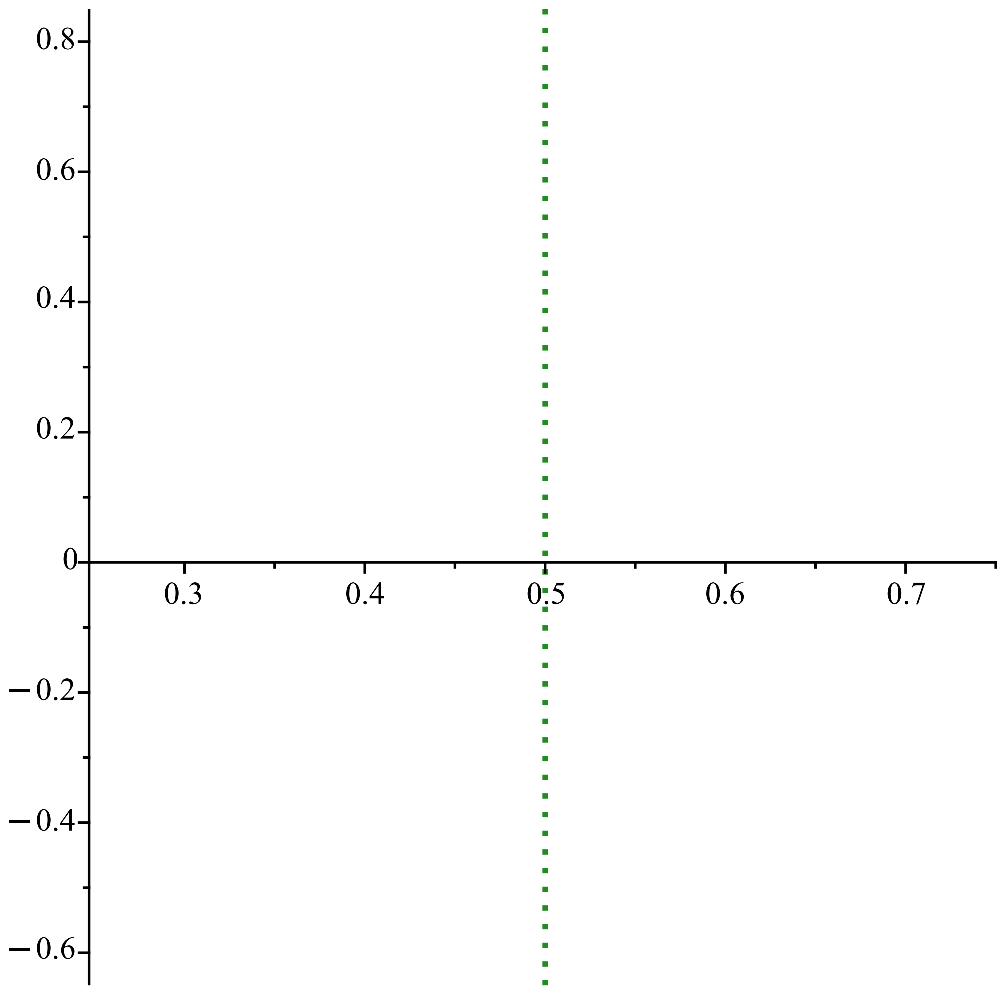


0
0.5
1
1.5
2
x
−0.2
−0.4
−0.6
−0.8
−1.0
1.2
psi(r/R) = N*(r-2R)

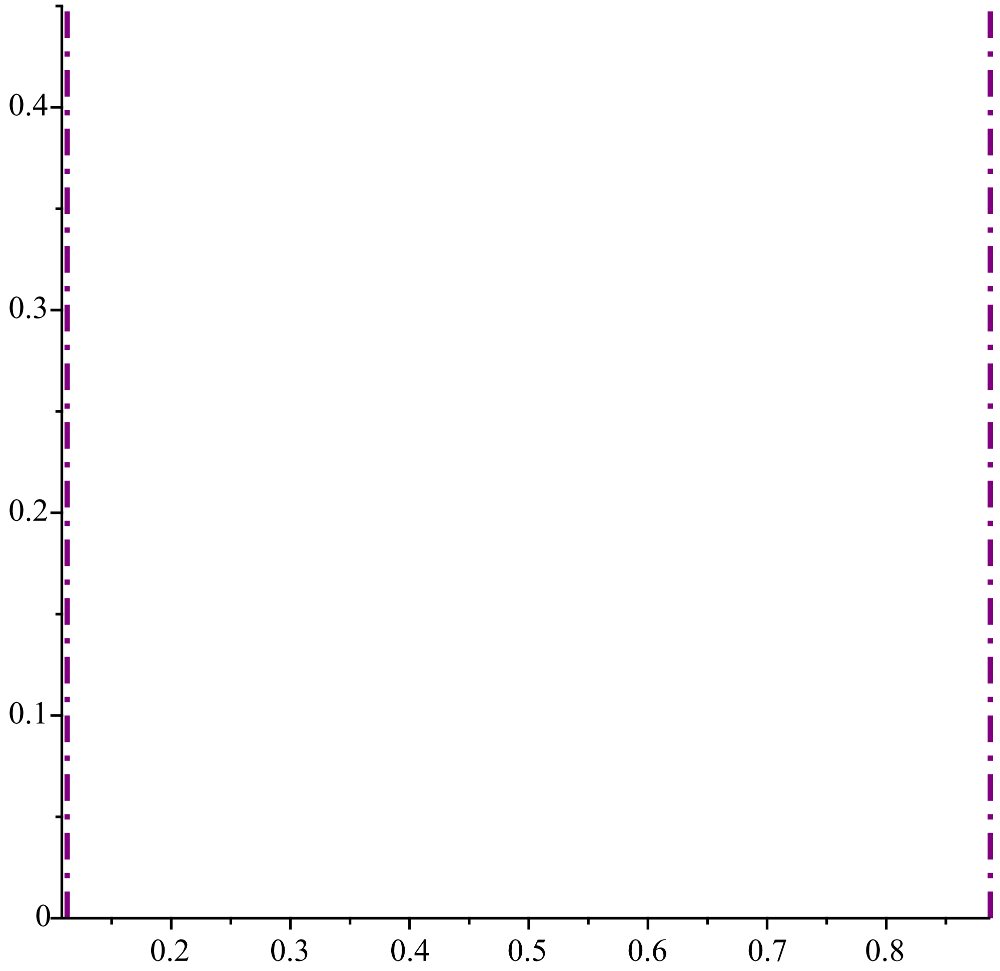


1.5
1
0.5
0
0
0.5
1
1.5
2
x
|psi|^2 = P(r/R) = N^2*(r-2R)^2

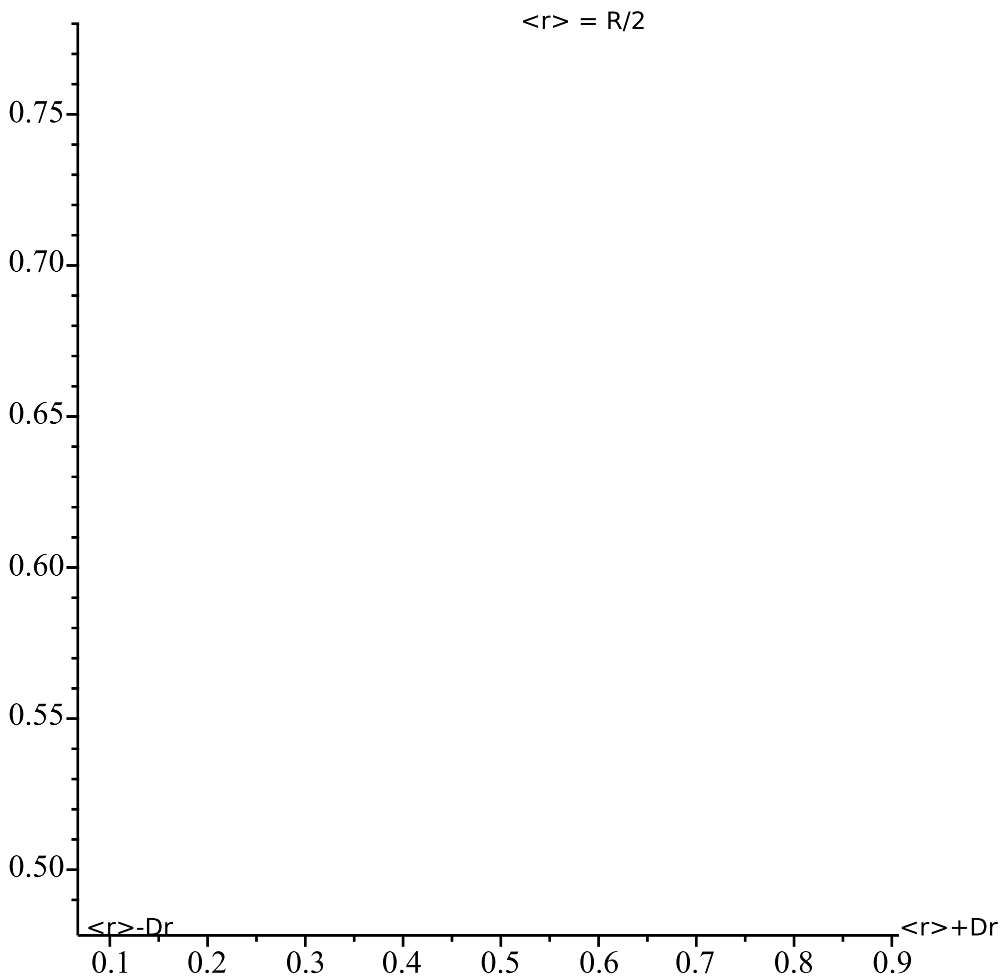




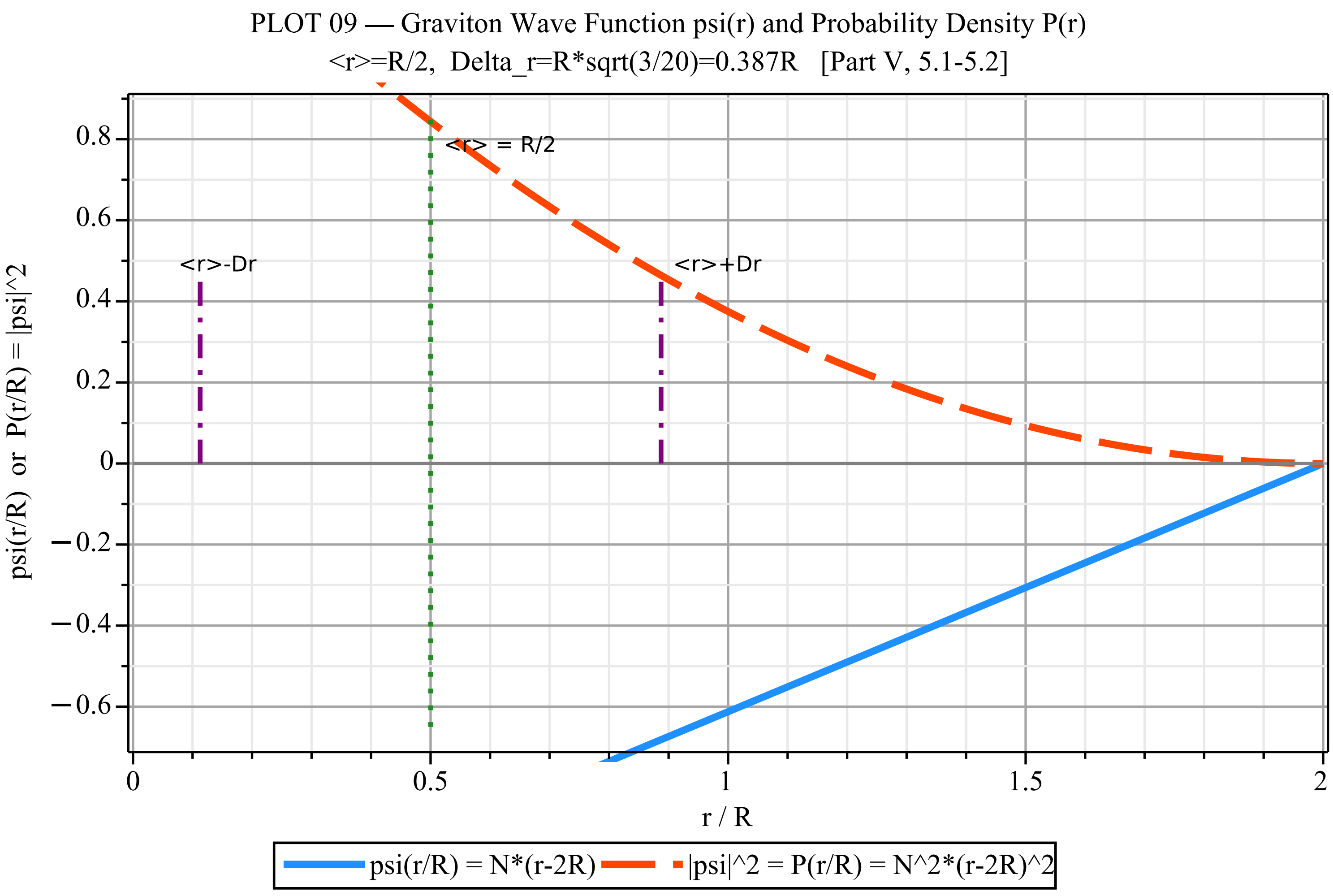
$Dr\_sc := 0.3872983346$



0.4
0.3
0.2
0.1
0
0.2
0.3
0.4
0.5
0.6
0.7
0.8

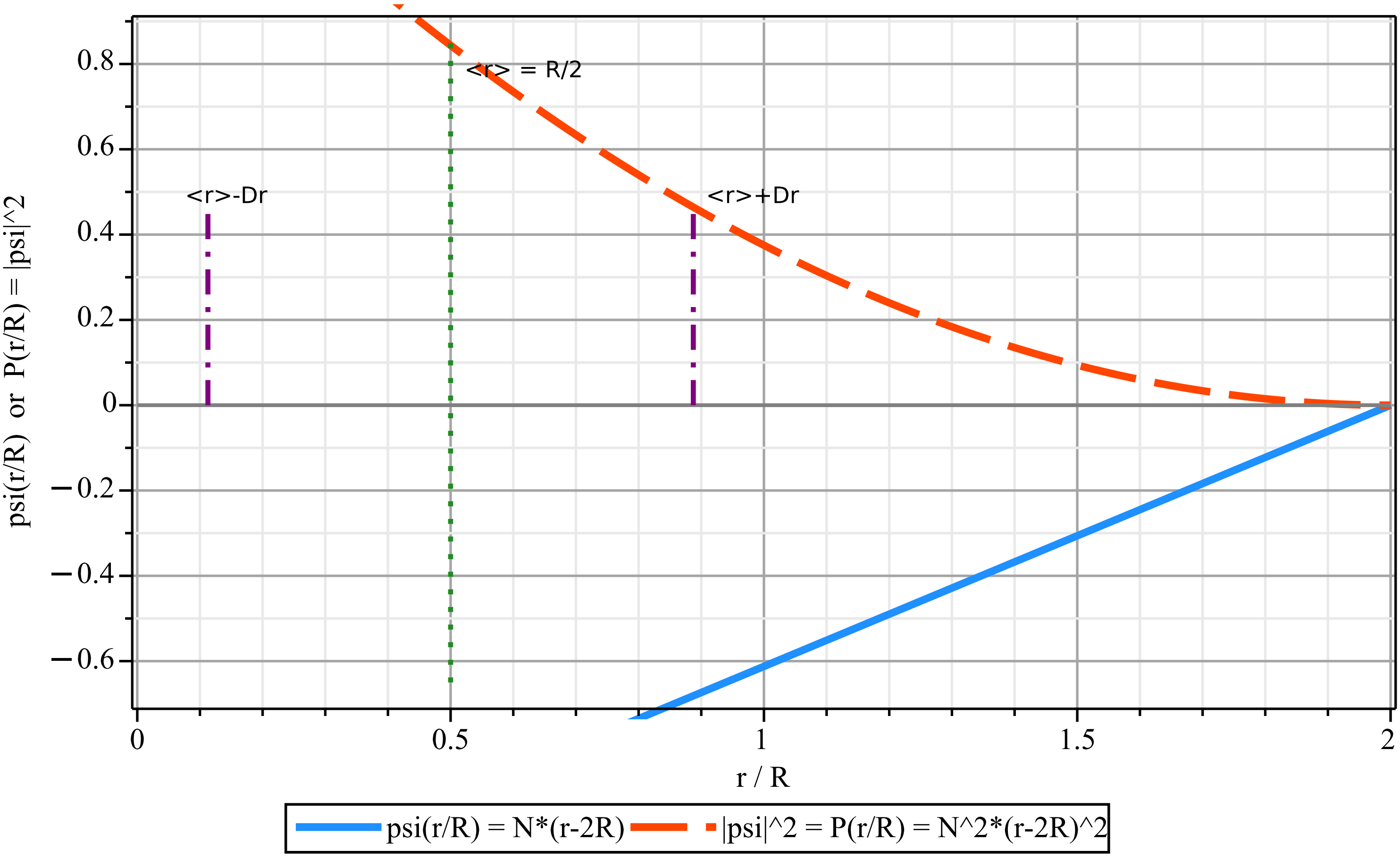


<r> = R/2
0.75
0.70
0.65
0.60
0.55
0.50
<r>-Dr
<r>+Dr
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9


PLOT 09 — Graviton Wave Function psi(r) and Probability Density P(r)
<r>=R/2, Delta_r=R*sqrt(3/20)=0.387R [Part V, 5.1-5.2]
psi(r/R) or P(r/R) = |psi|^2
r / R
<r> = R/2
<r>-Dr
<r>+Dr
psi(r/R) = N*(r-2R)
|psi|^2 = P(r/R) = N^2*(r-2R)^2

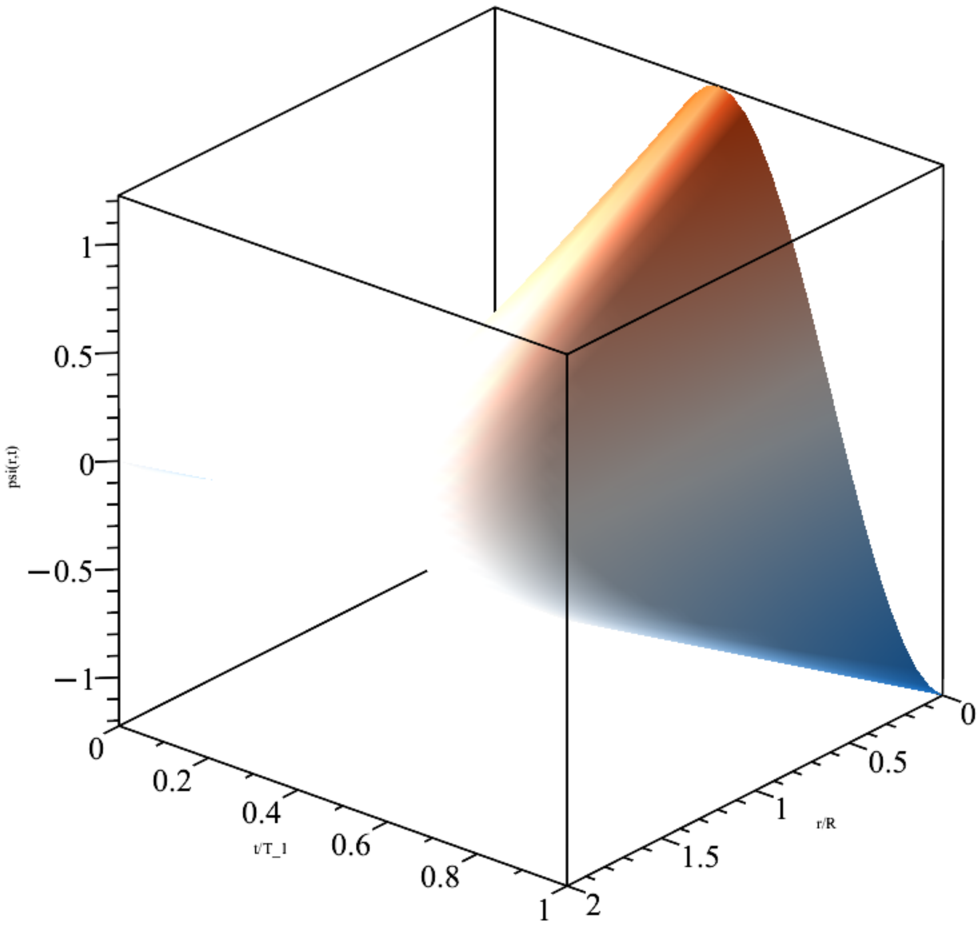
"PLOT 09 done."

PLOT 09 — Graviton Wave Function psi(r) and Probability Density P(r)
<r>=R/2, Delta_r=R*sqrt(3/20)=0.387R [Part V, 5.1-5.2]



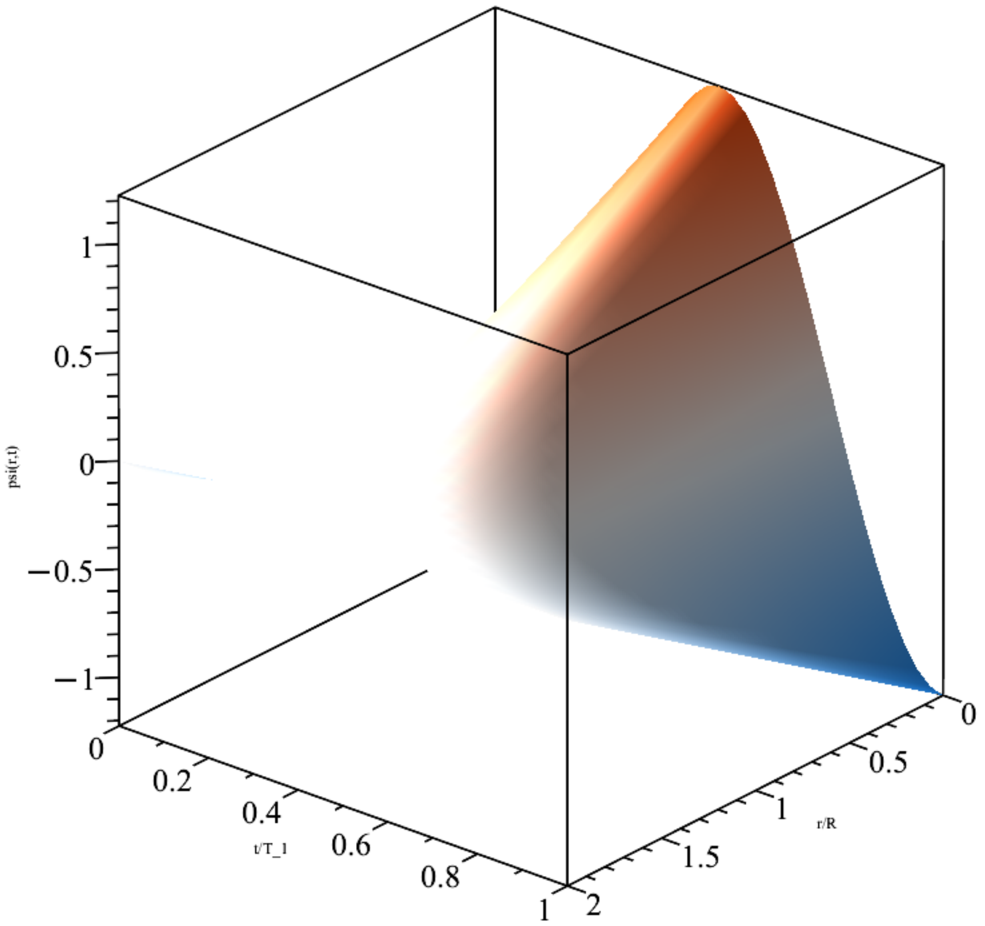

```
==============================================================
 PLOT 10: Spacetime Evolution psi(r,t) — 3D surface
==============================================================
```

PLOT 10 — Graviton Field psi(r,t) = N*(r-2R)*cos(omega1*t) [3D]
x=r/R in [0,2], tau=t/T_1 in [0,1]   [Part V, 5.3]



"PLOT 10 done."

PLOT 10 — Graviton Field psi(r,t) = N*(r-2R)*cos(omega1*t) [3D]
x=r/R in [0,2], tau=t/T_1 in [0,1]   [Part V, 5.3]

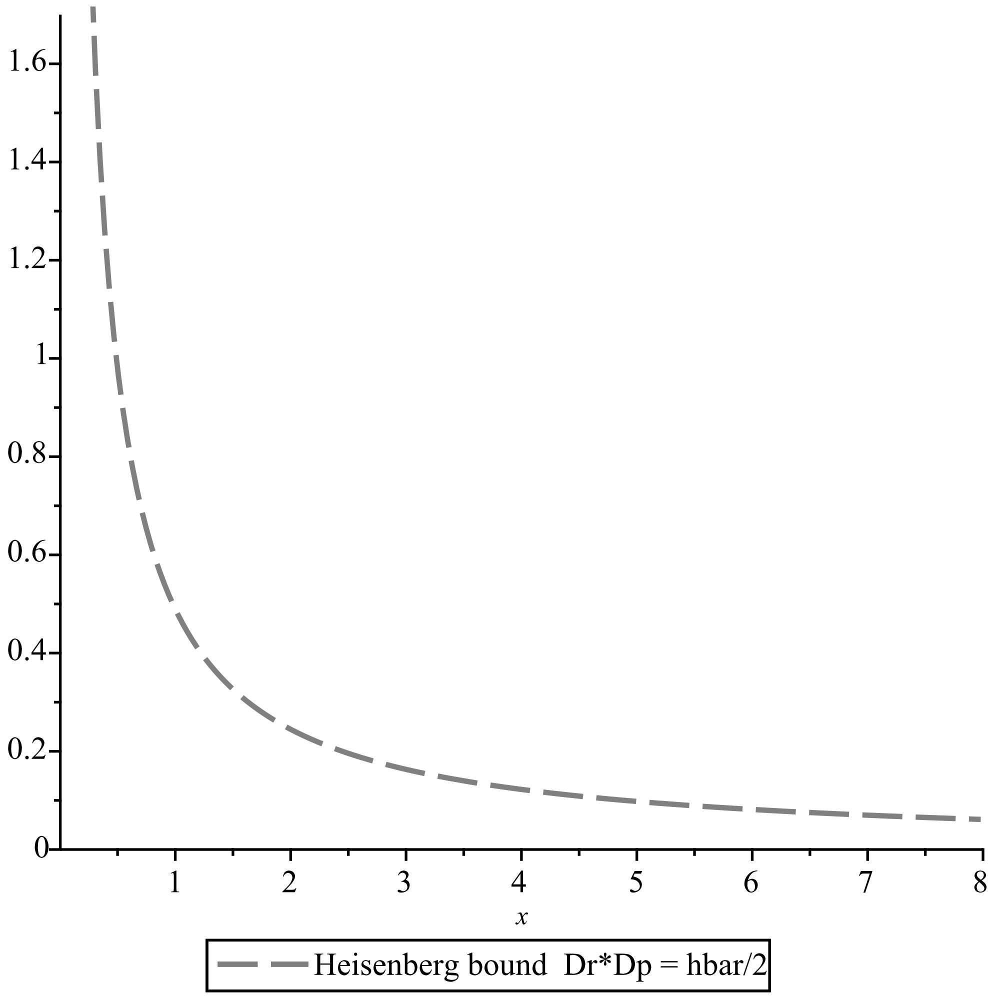


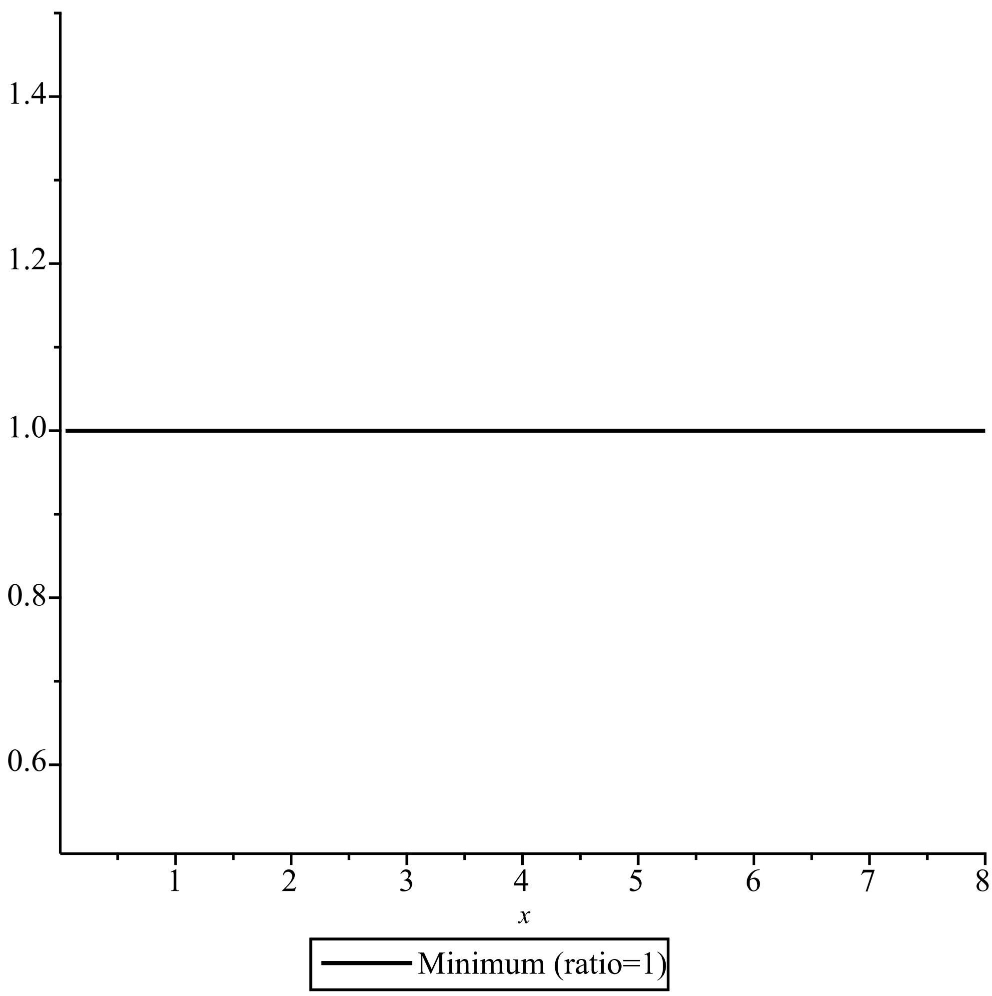
==
PLOT 11: Heisenberg Phase-Space Diagram
==

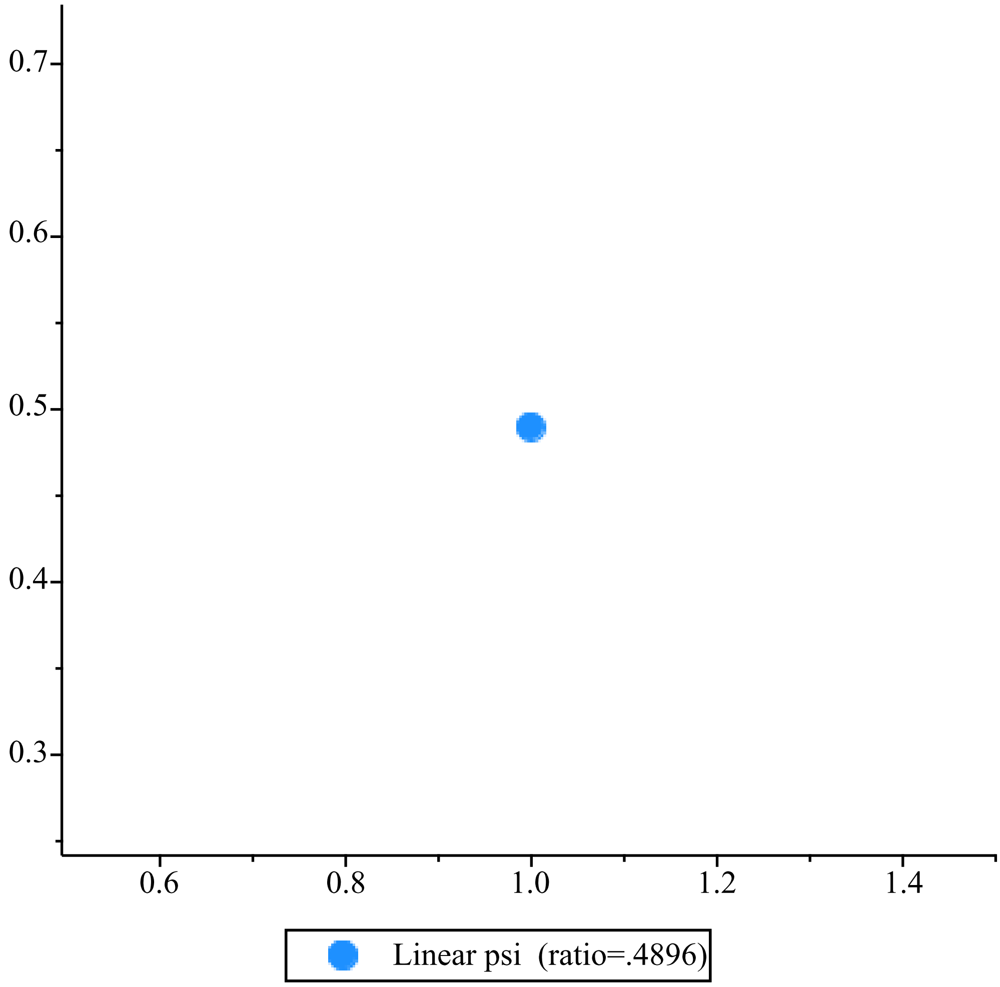
$$ratio\_lin := 0.4895503030$$

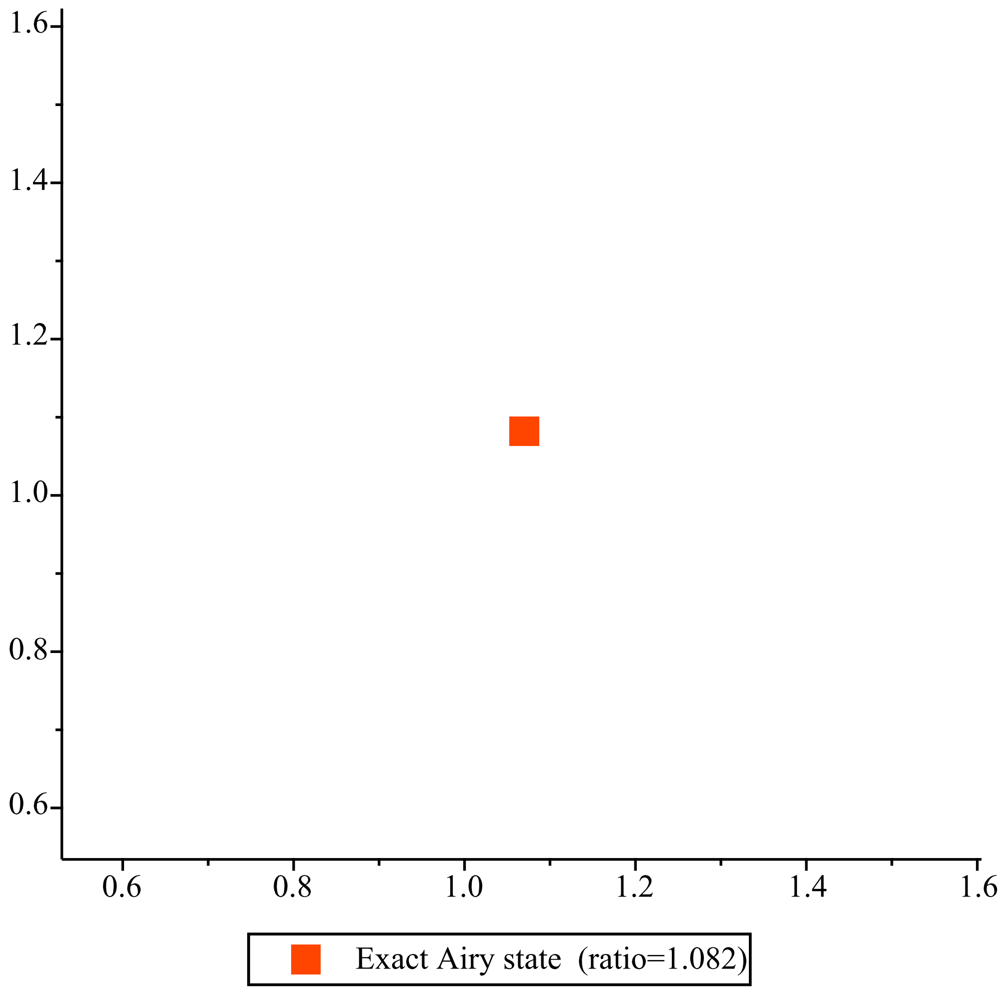
$$ratio\_Airy := 1.082$$

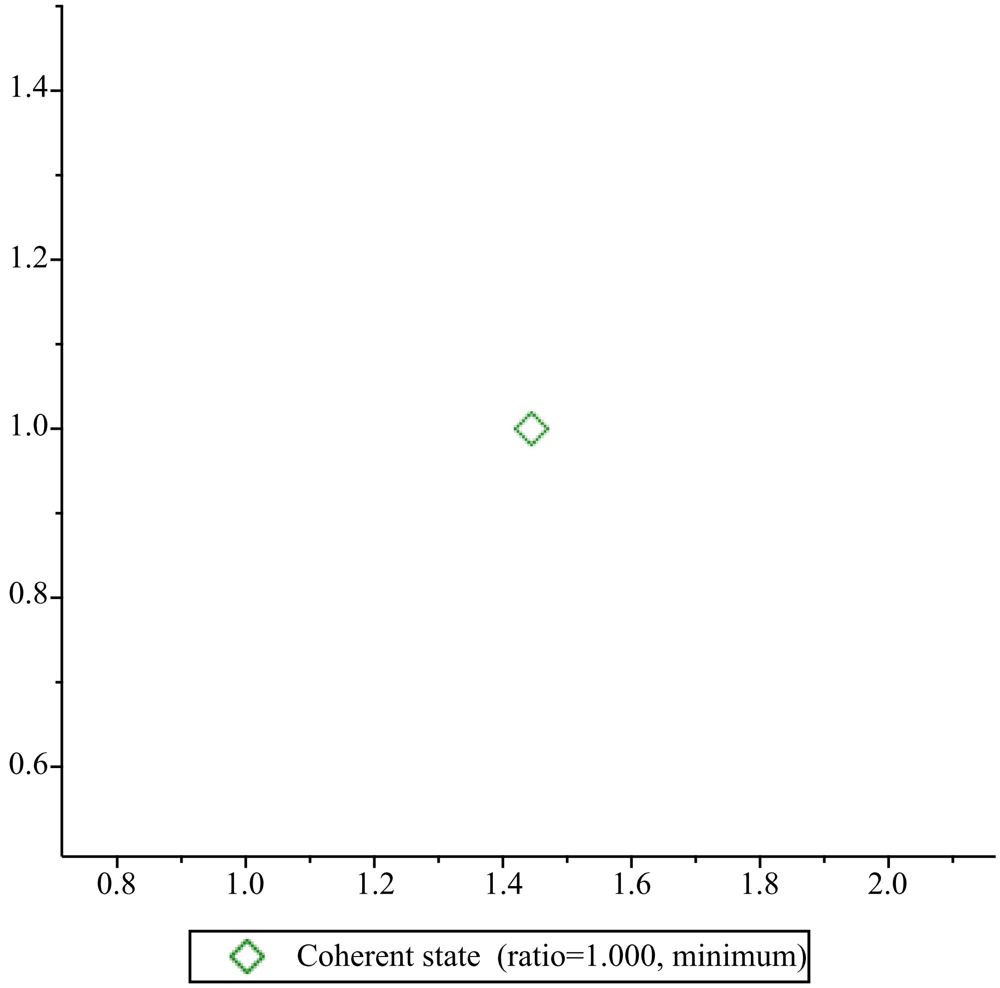
$$sigma\_norm := 1.444400661$$


1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
x
Heisenberg bound Dr*Dp = hbar/2

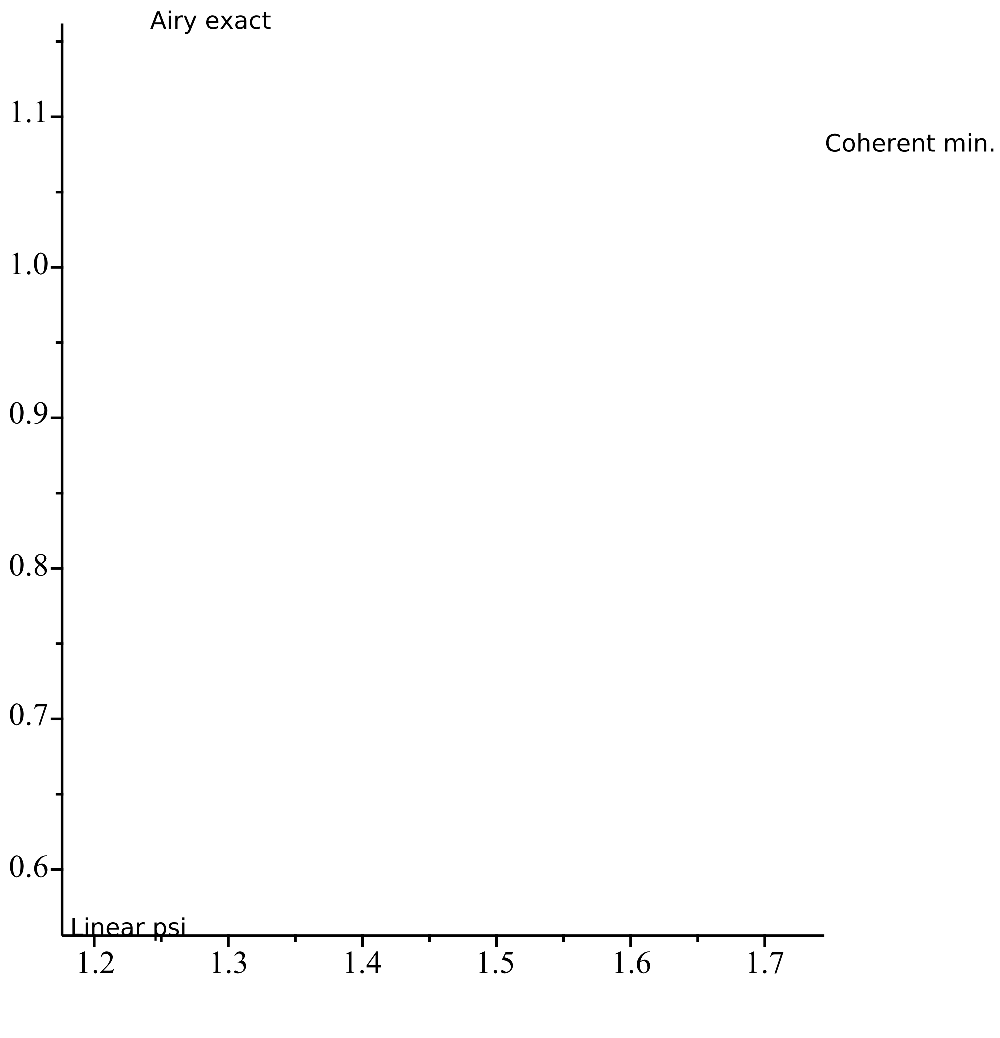

1.4
1.2
1.0
0.8
0.6
1
2
3
4
5
6
7
8
x
Minimum (ratio=1)


0.7
0.6
0.5
0.4
0.3
0.6
0.8
1.0
1.2
1.4
Linear psi (ratio=.4896)

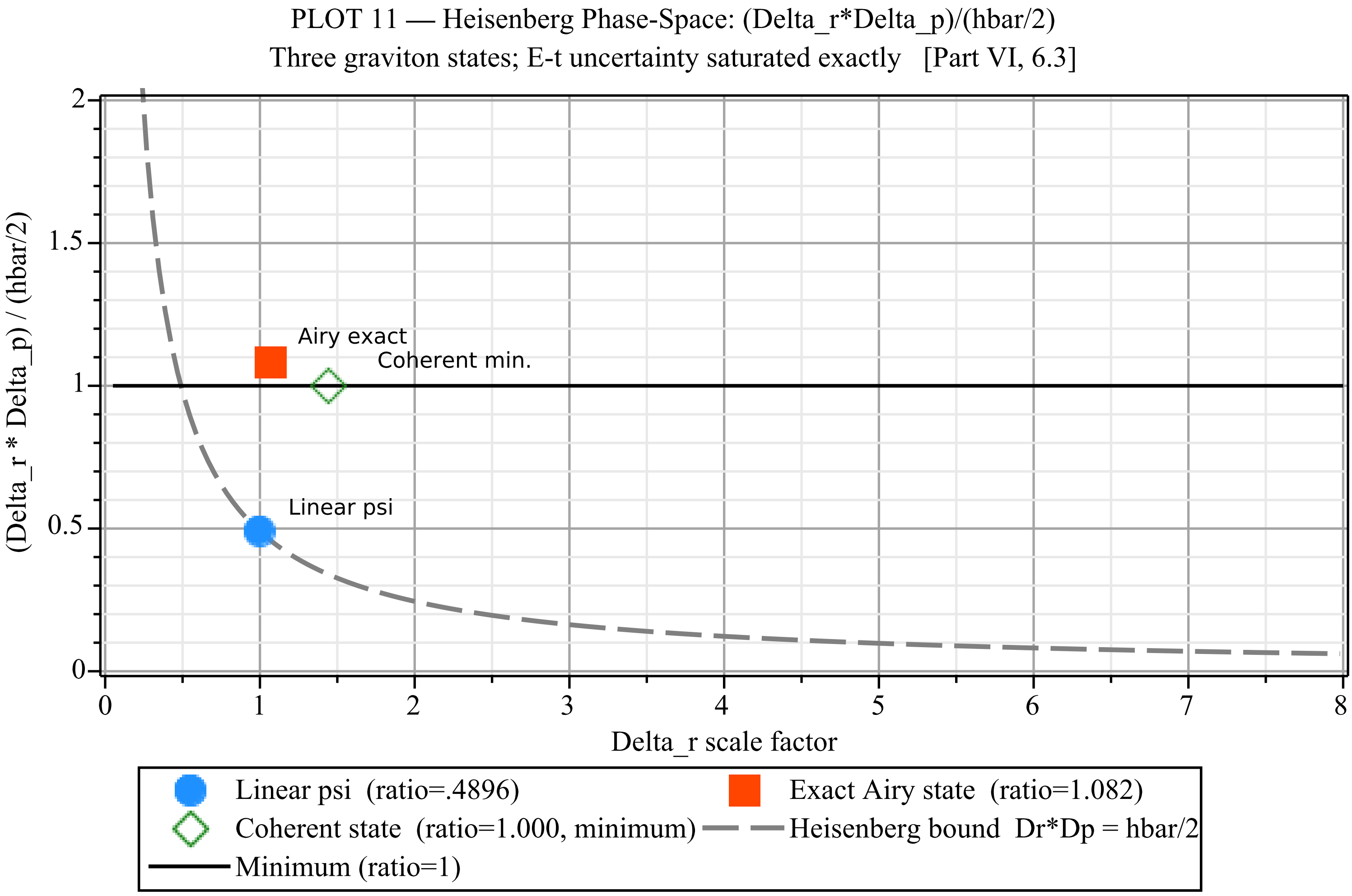

1.6
1.4
1.2
1.0
0.8
0.6
0.6
0.8
1.0
1.2
1.4
1.6
Exact Airy state (ratio=1.082)



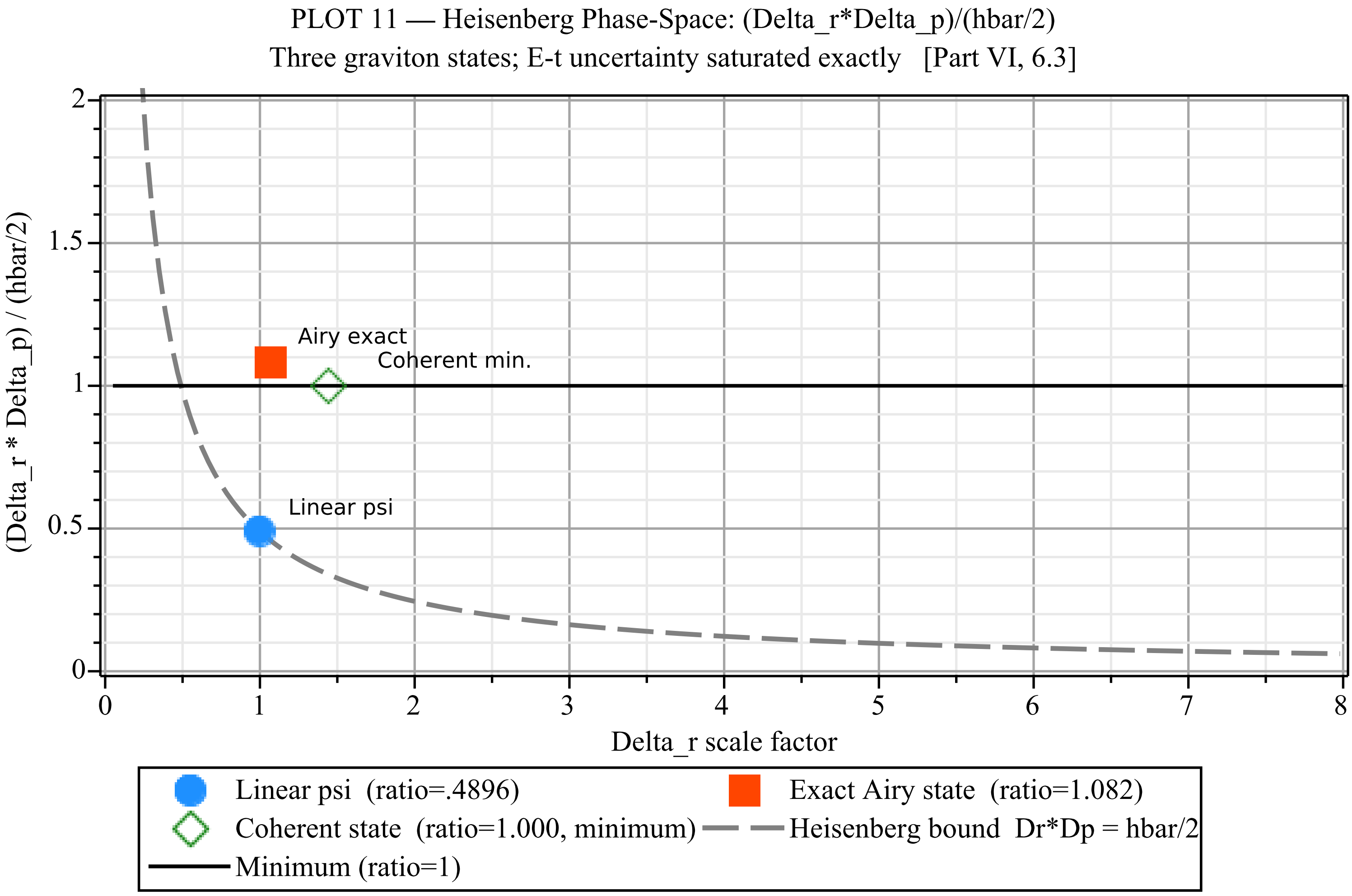
Coherent state (ratio=1.000, minimum)



Airy exact
Coherent min.
Linear psi


PLOT 11 — Heisenberg Phase-Space: (Delta_r*Delta_p)/(hbar/2)
Three graviton states; E-t uncertainty saturated exactly [Part VI, 6.3]
(Delta_r * Delta_p) / (hbar/2)
Delta_r scale factor
Airy exact
Coherent min.
Linear psi
0
0.5
1
1.5
2
0
1
2
3
4
5
6
7
8
Linear psi (ratio=.4896)
Exact Airy state (ratio=1.082)
Coherent state (ratio=1.000, minimum)
Heisenberg bound Dr*Dp = hbar/2
Minimum (ratio=1)

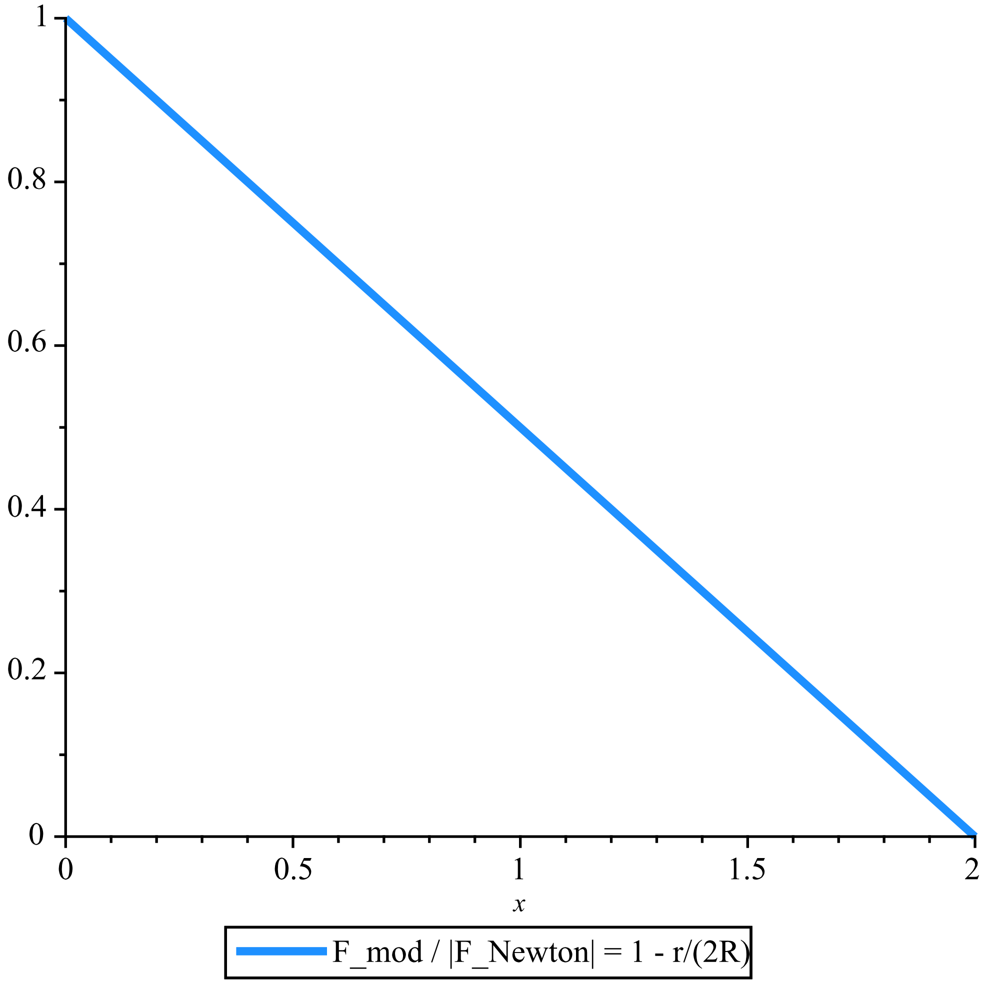
"PLOT 11 done."



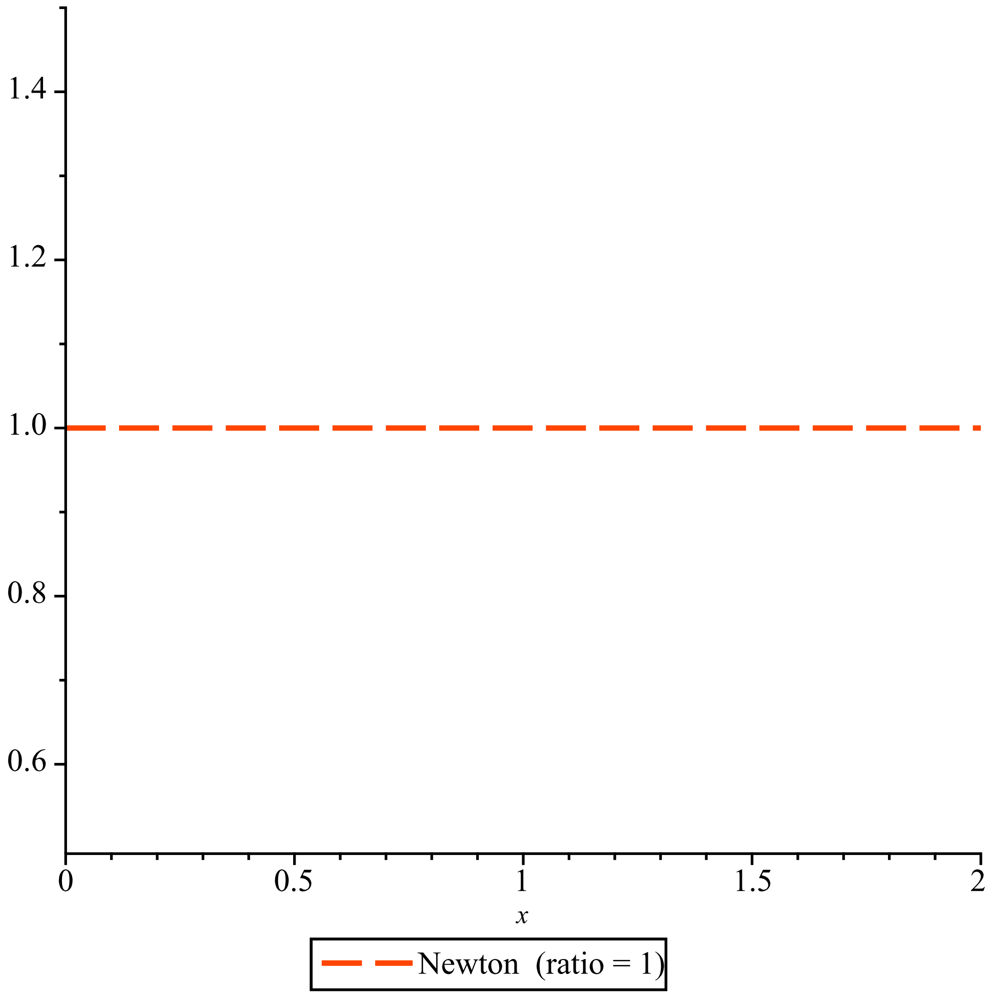
==

PLOT 12: Quantum-Classical Bridge

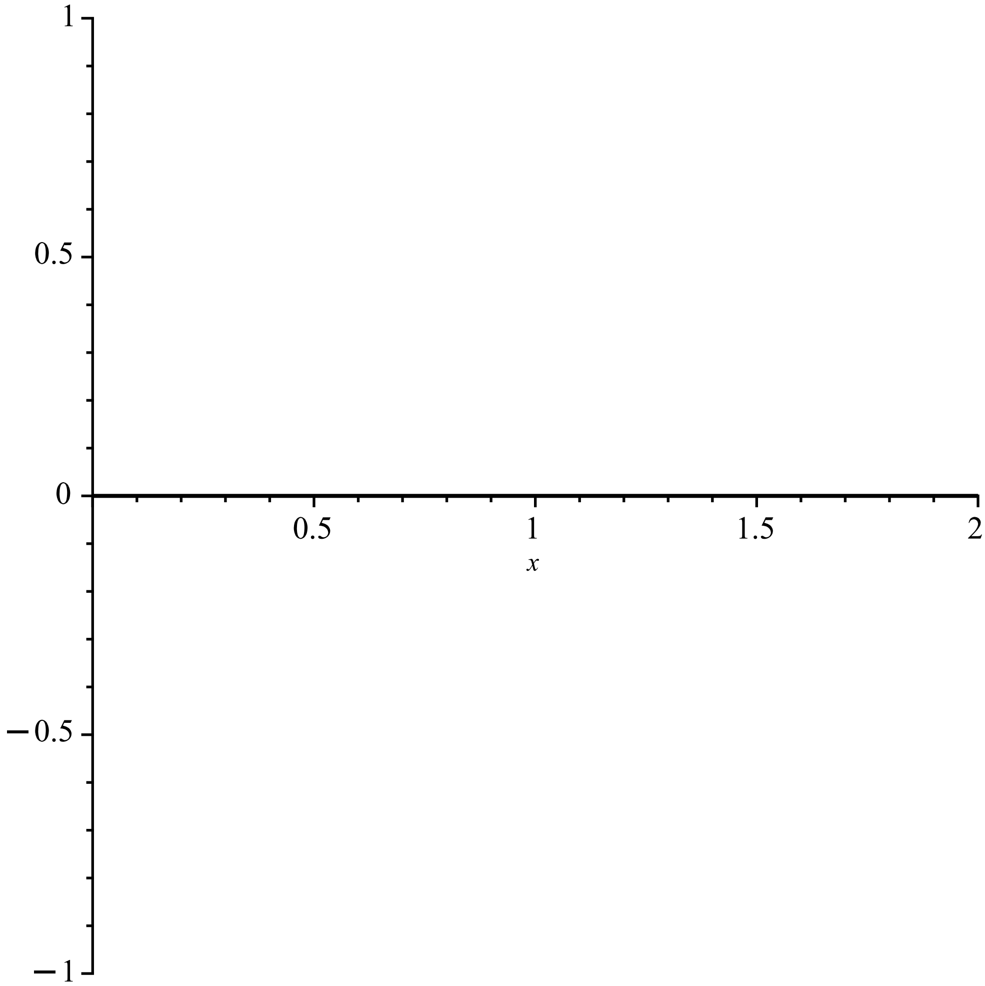
==



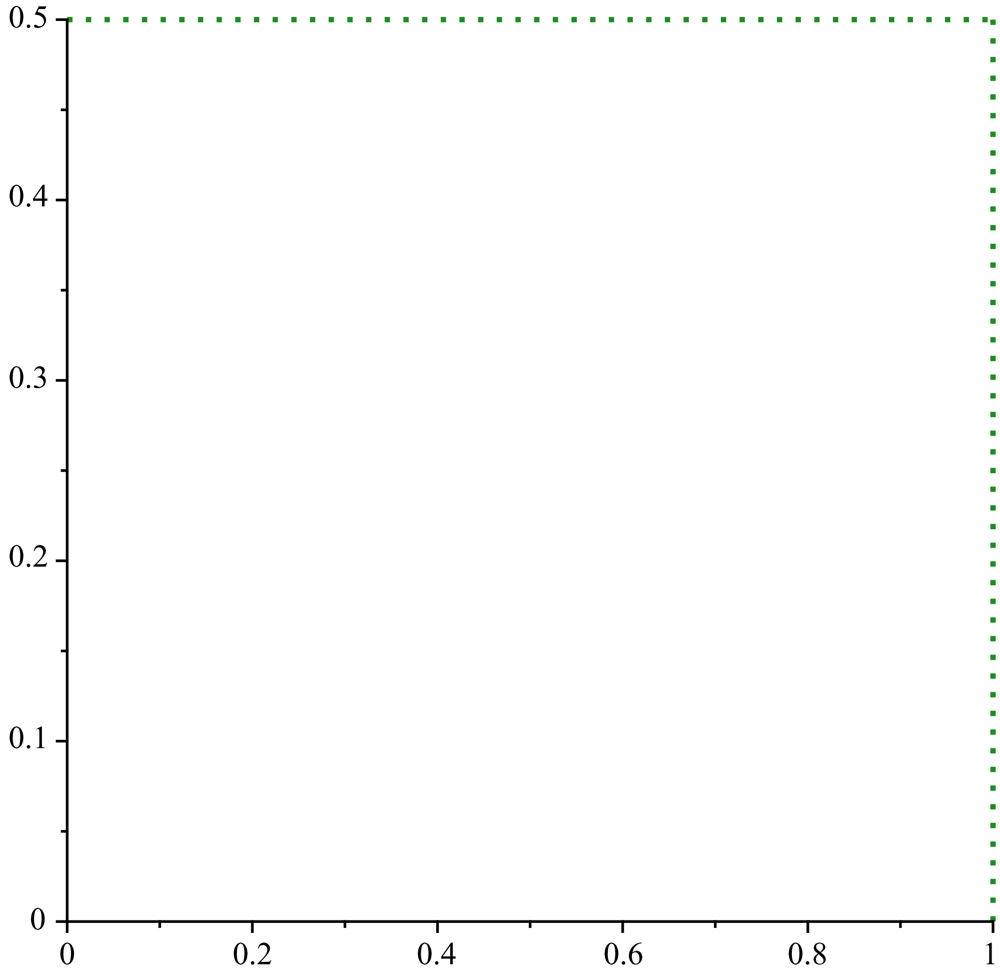
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
x
F_mod / |F_Newton| = 1 - r/(2R)



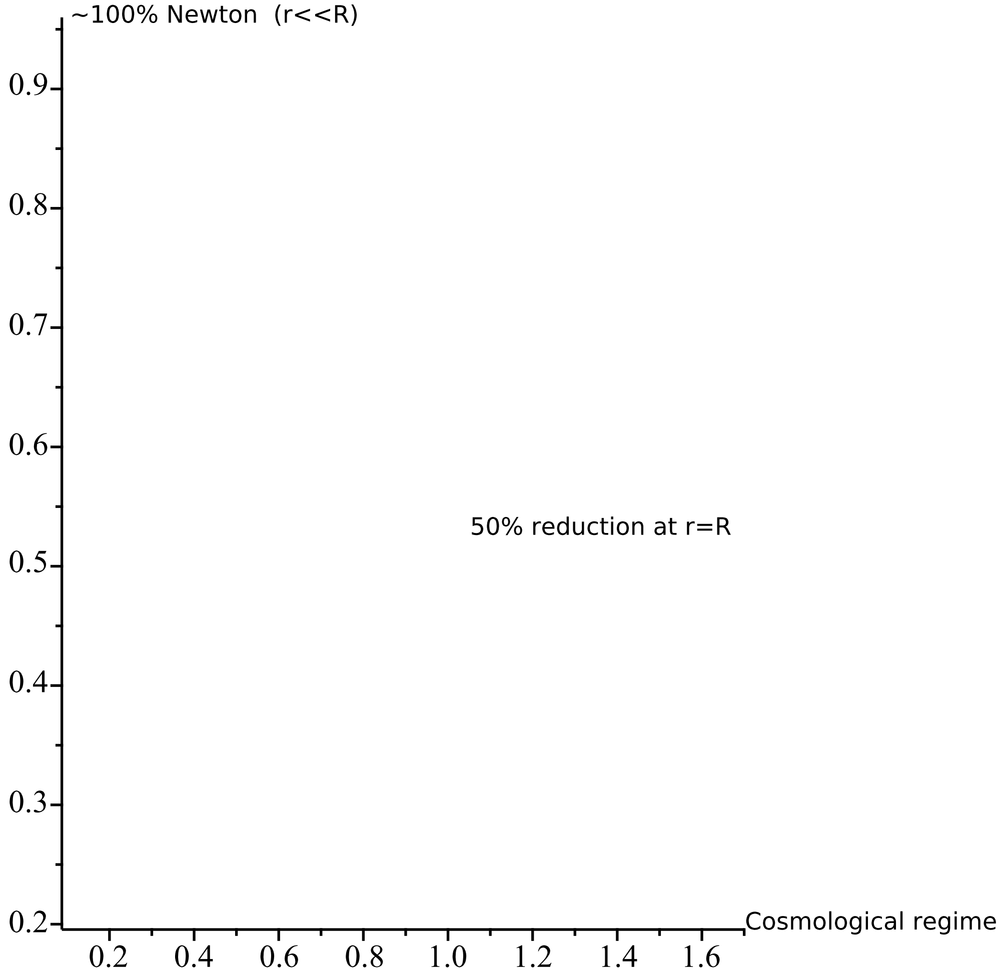
1.4
1.2
1.0
0.8
0.6
0
0.5
1
1.5
2
x
Newton (ratio = 1)



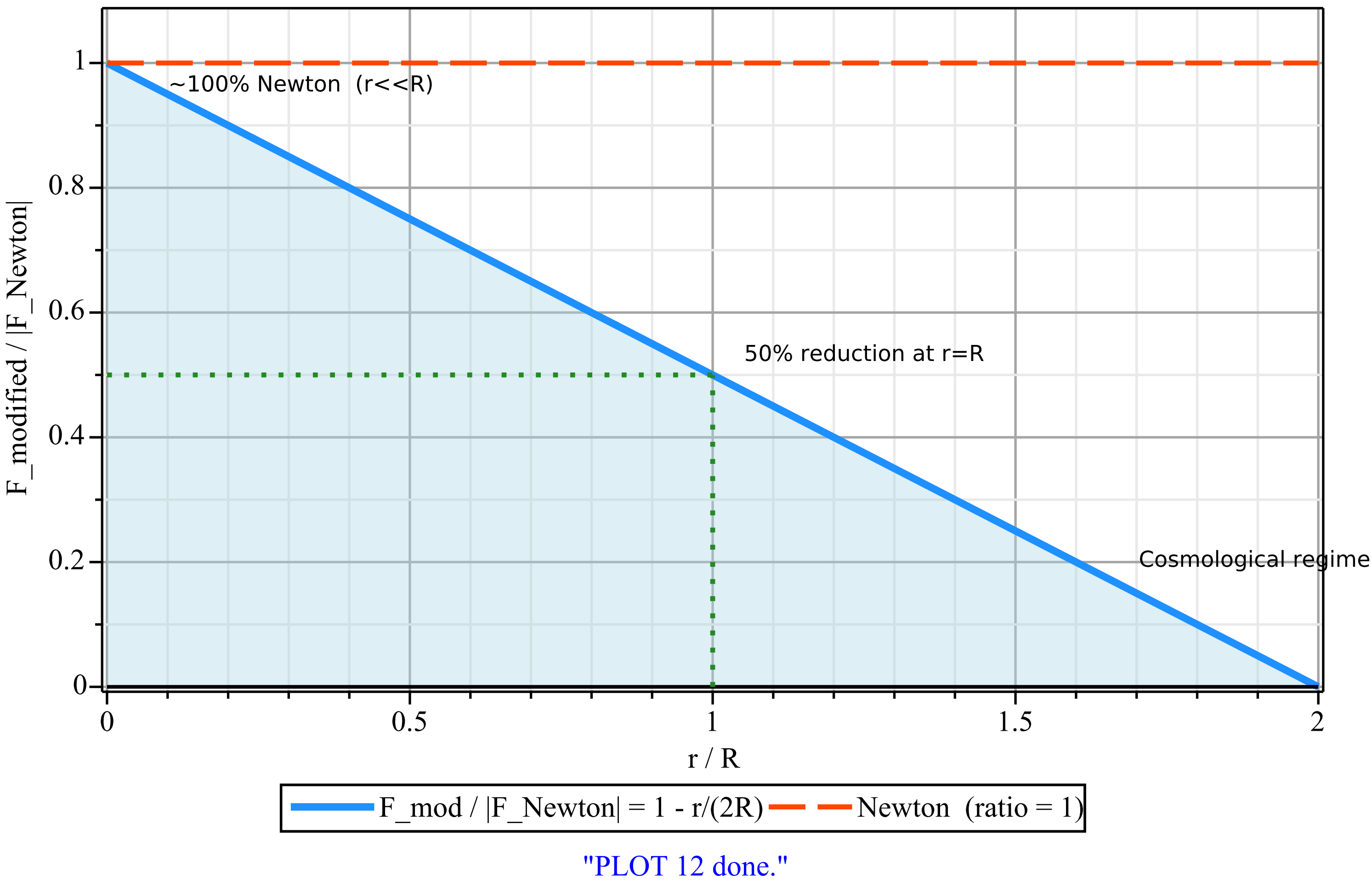
1
0.5
0
−0.5
−1
0.5
1
1.5
2
x



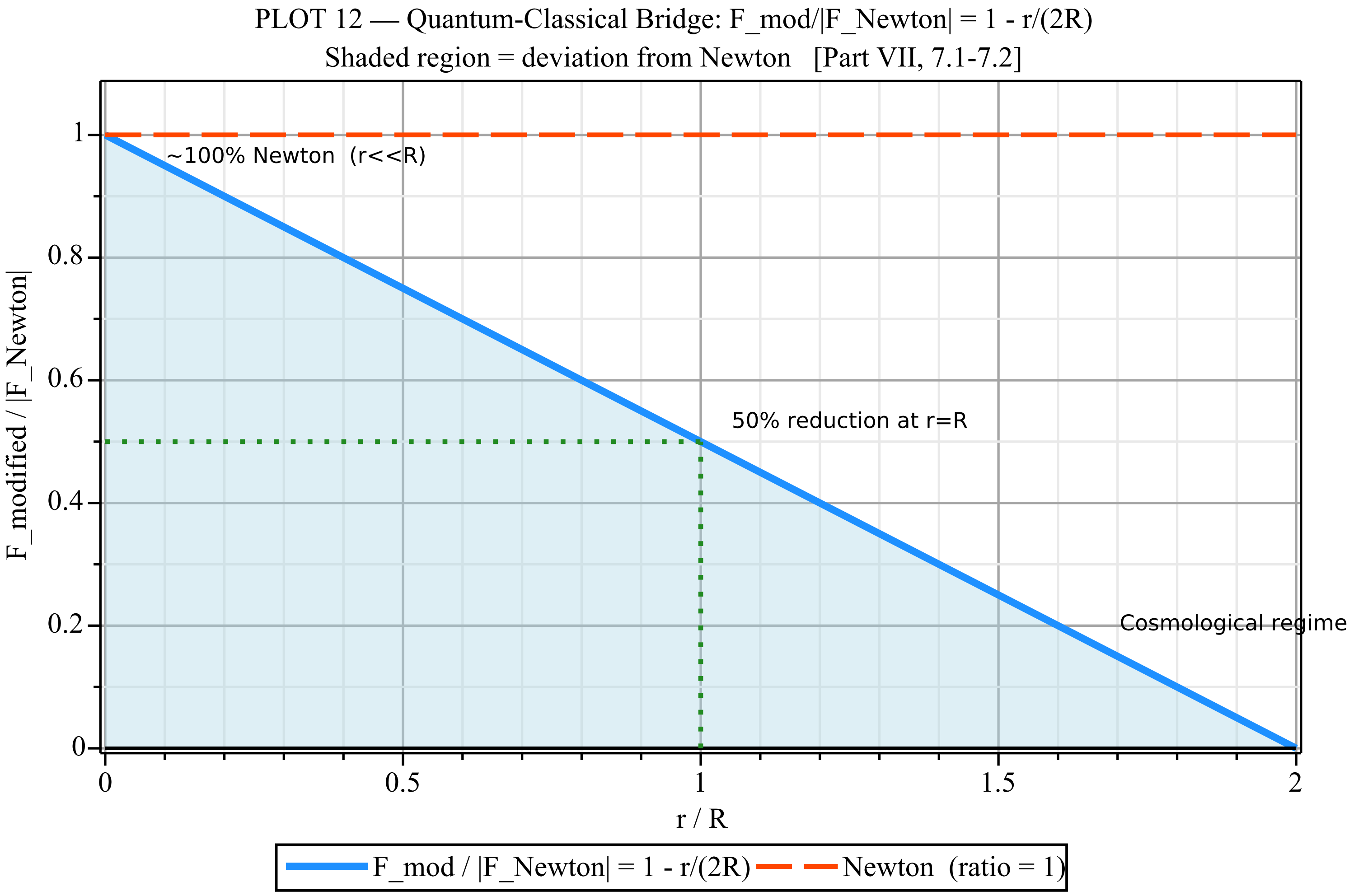
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1



PLOT 12 — Quantum-Classical Bridge: F_mod/|F_Newton| = 1 - r/(2R)
Shaded region = deviation from Newton [Part VII, 7.1-7.2]



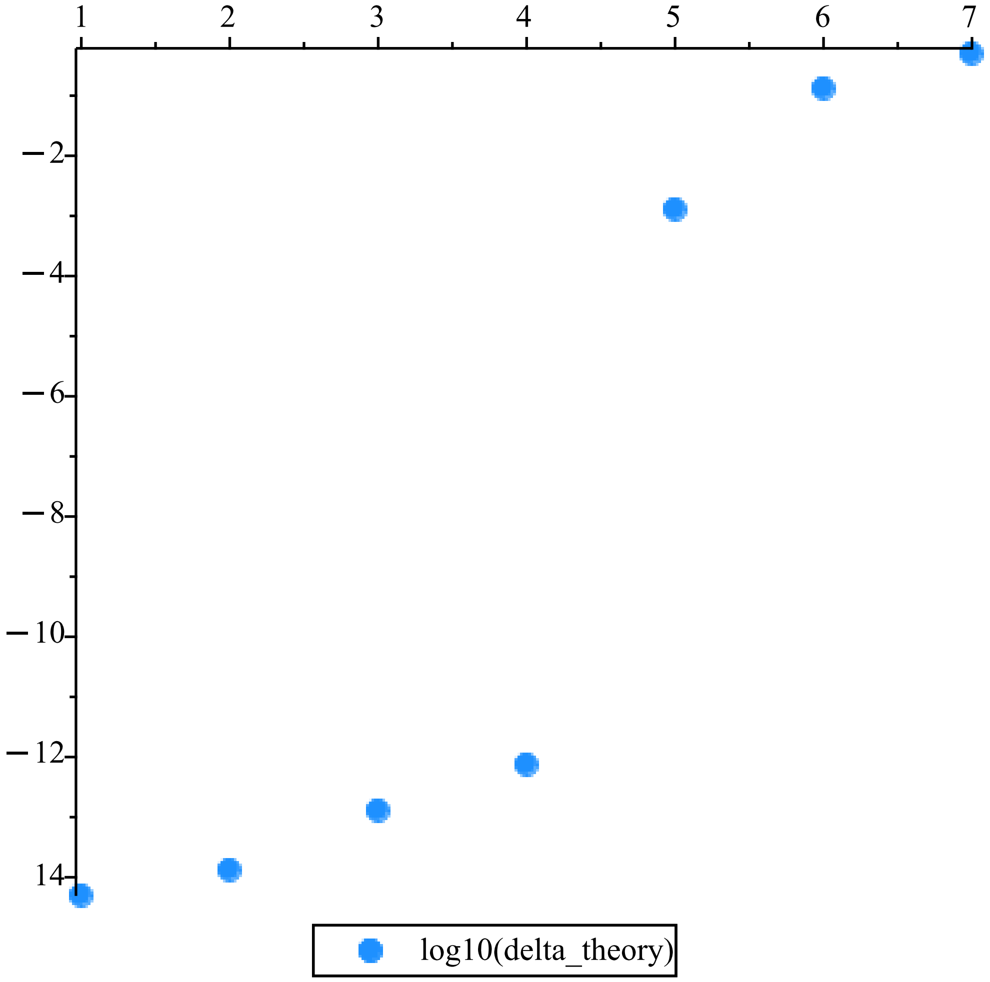
"PLOT 12 done."

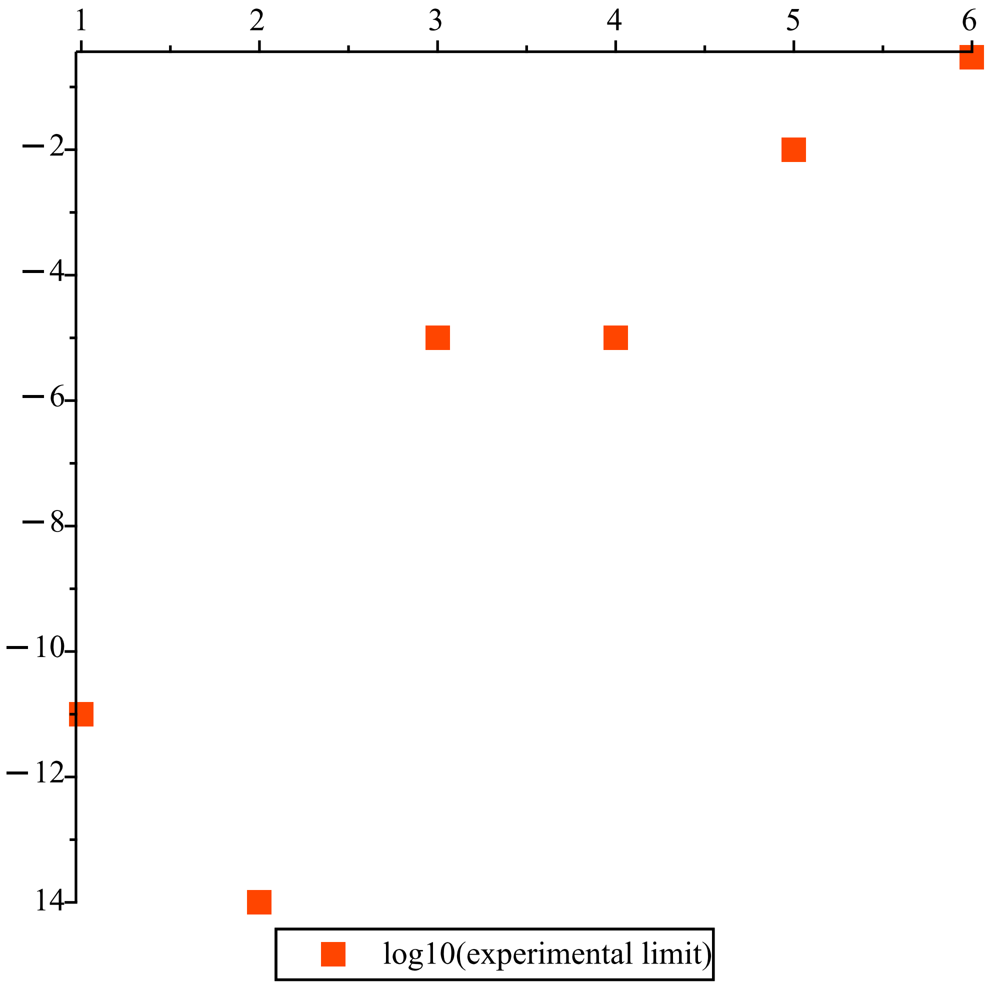


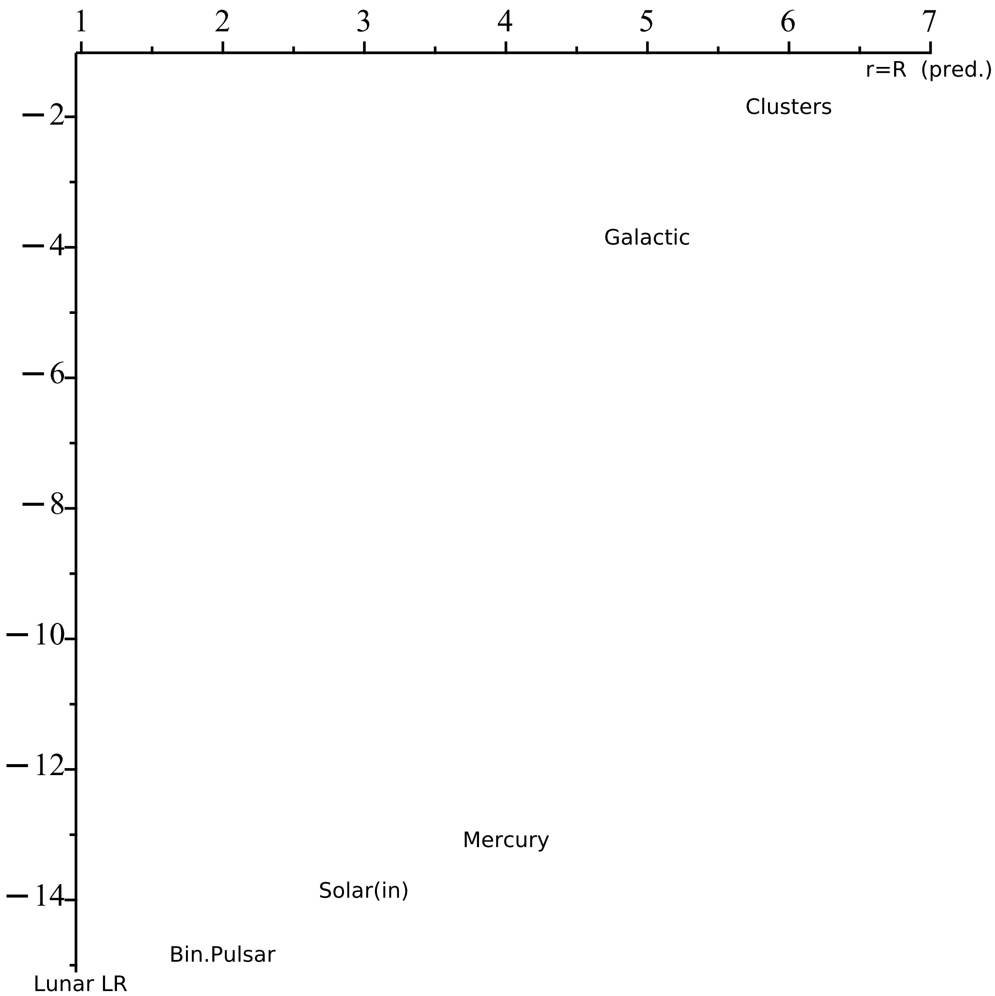
==

PLOT 13: Solar System Precision Tests

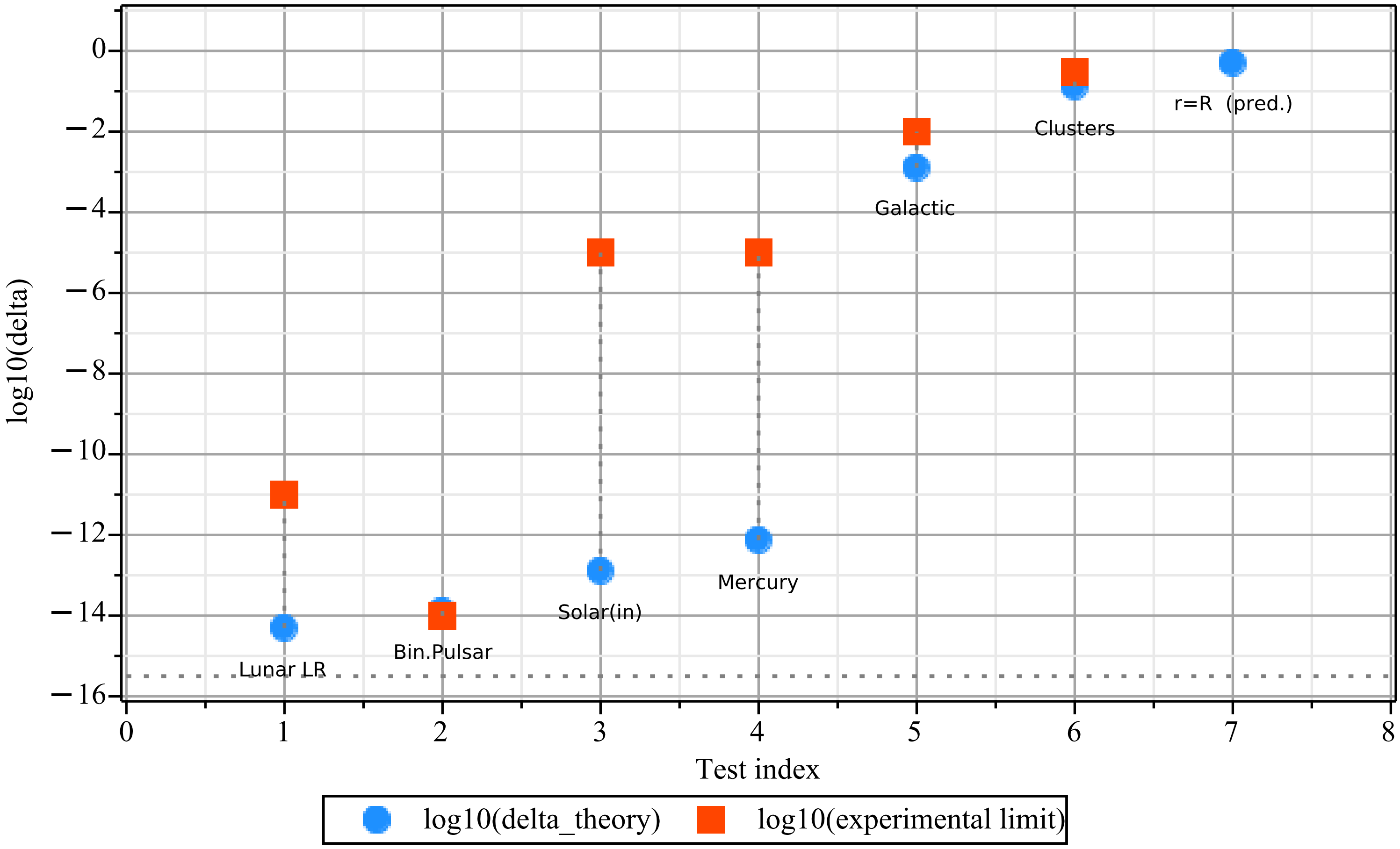
==


1 2 3 4 5 6 7
−2
−4
−6
−8
−10
−12
14
log10(delta_theory)


1 2 3 4 5 6
−2
−4
−6
−8
−10
−12
14
log10(experimental limit)

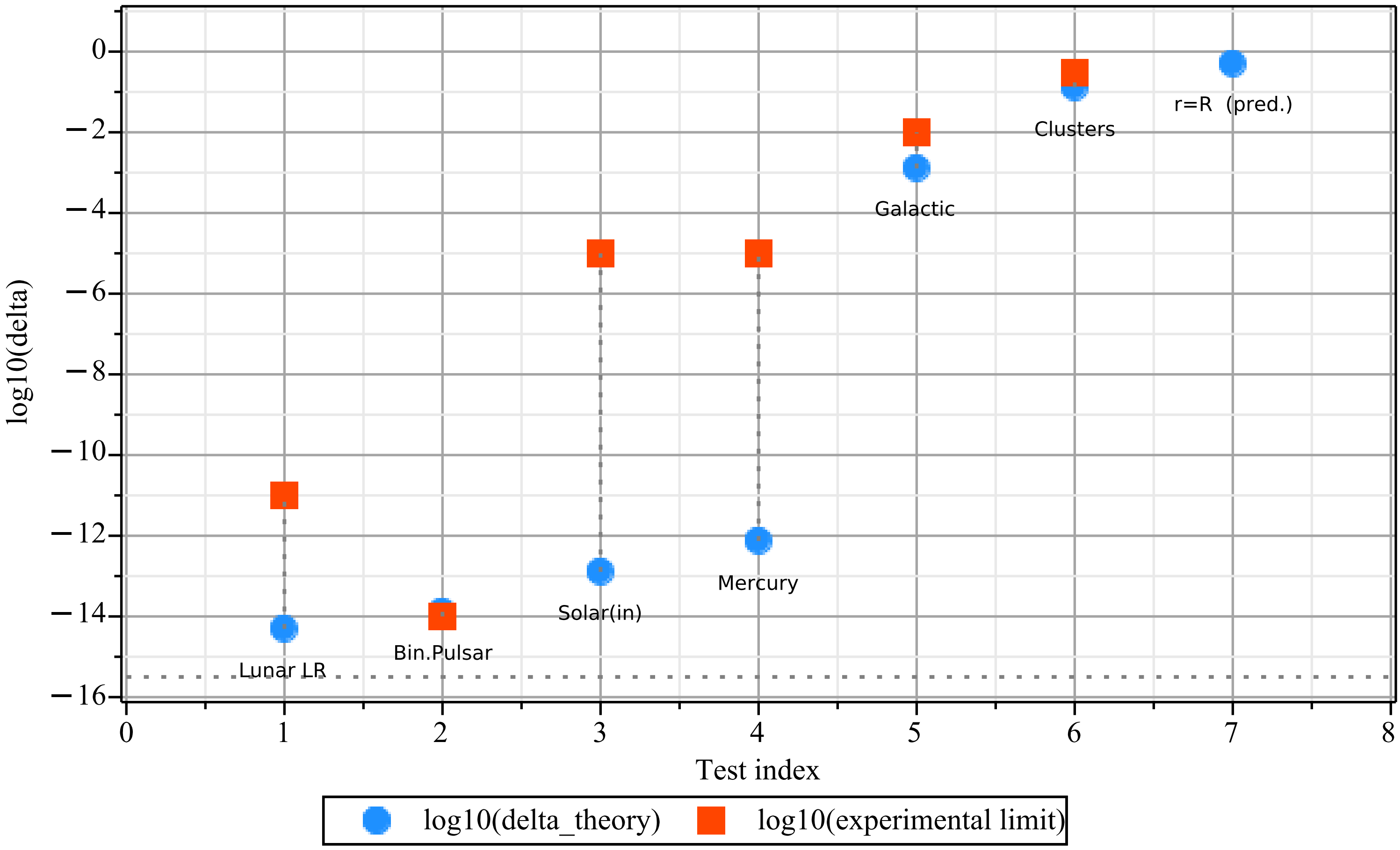


PLOT 13 — Solar System and Astronomical Precision Tests
Blue=theory delta(r), Red=experimental limit [log10 scale] [Part VIII]



"PLOT 13 done."

PLOT 13 — Solar System and Astronomical Precision Tests
Blue=theory delta(r), Red=experimental limit [log10 scale] [Part VIII]

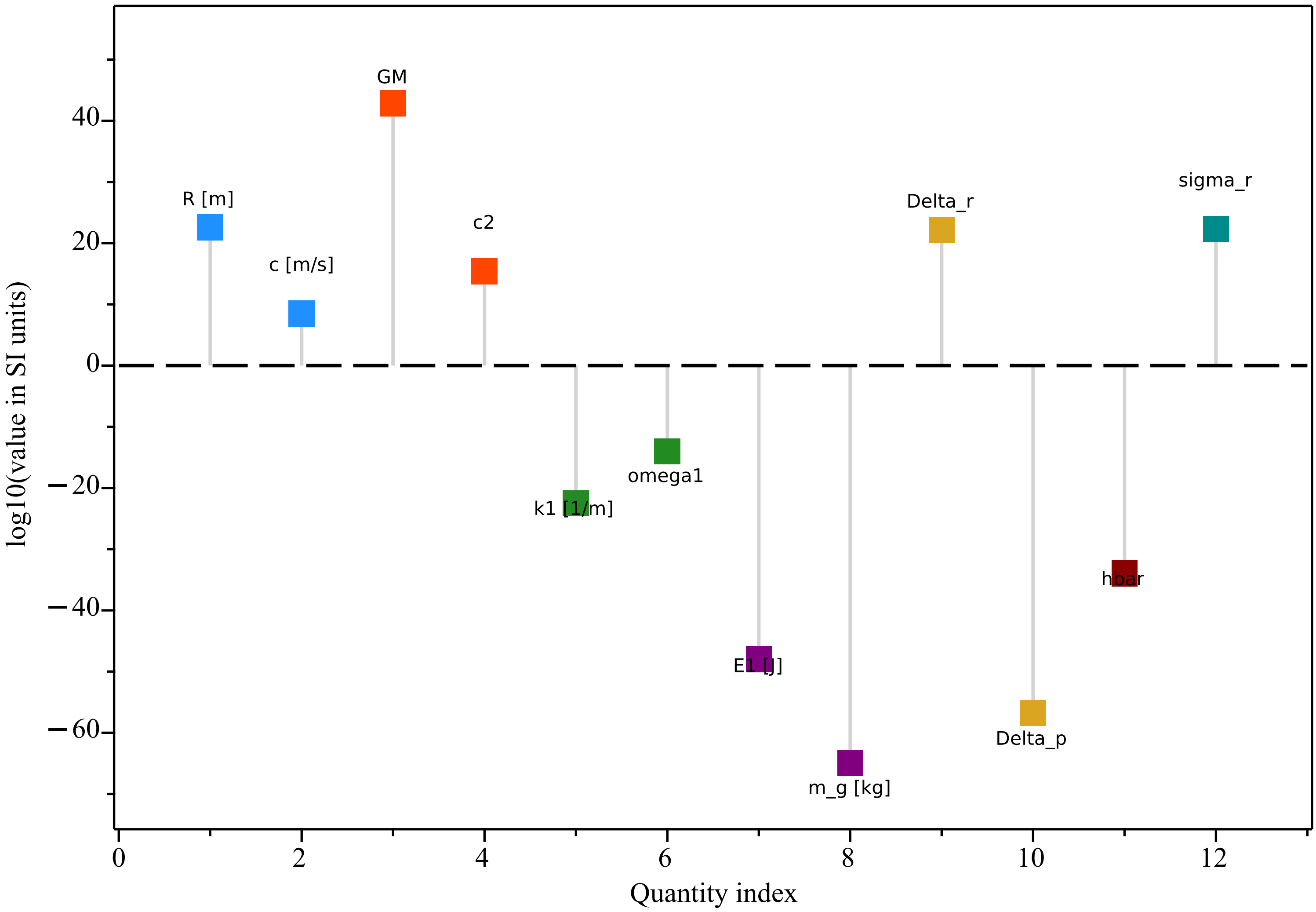



```
==============================================================
 FULL SUMMARY TABLE
==============================================================
Quantity                                                     Value
Unit
------------------------------------------------------------------------------
--
R (cosmological distance)                             3.922800e+22
m
c (speed of light)                                    2.997925e+08
m/s
G (Newton constant)                                   6.674000e-11    N
m^2/kg^2
hbar                                                  1.054570e-34          J
s
GM                                                    6.674000e+42
m^3/s^2
c2 = 2pi c^2 GM/R^2                                   2.449149e+15
m^3/s^2
c1                                                      0  (exact)
m^2/s^2
N (normalization)                                     7.881727e-35      m^
(-3/2)
------------------------------------------------------------------------------
```

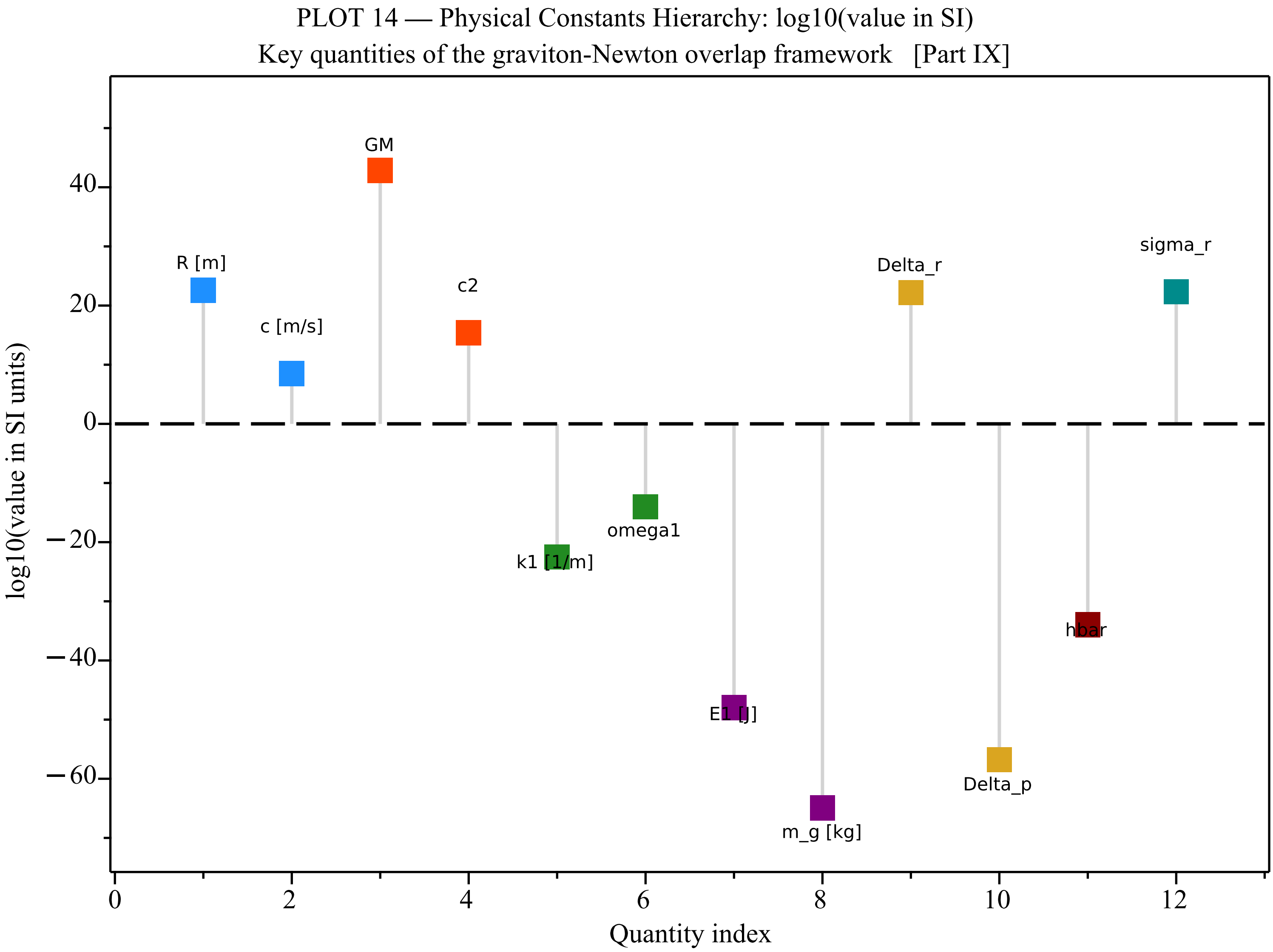
```
--
k1 (fund. wave vector)                                   3.222223e-23
m^-1
k2                                                       7.449328e-23
m^-1
k3                                                       1.169051e-22
m^-1
omega1                                                   9.659980e-15
rad/s
T1 (fund. period)                                        6.504346e+14
s
T1                                                          20.609461
Myr
E1 (fund. graviton energy)                               1.018713e-48
J
m_g (graviton mass bound)                                1.133471e-65
kg
------------------------------------------------------------------------------
--
<r> = R/2                                                1.961400e+22
m
Delta_r = R*sqrt(3/20)                                   1.519294e+22
m
Delta_r / R                                                  0.387298

Delta_p = hbar*k1/2                                      1.699030e-57         kg
m/s
Delta_r * Delta_p                                        2.581325e-35            J
s
hbar/2                                                   5.272850e-35            J
s
Ratio (pos-mom)/(hbar/2)                                     0.489550

Ratio Airy exact                                             1.082000

------------------------------------------------------------------------------
--
Delta_E = hbar*omega1/2                                  5.093563e-49
J
Delta_t = 1/omega1                                       1.035199e+14
s
Delta_t                                                      3.280098
Myr
Delta_E * Delta_t                                        5.272850e-35            J
s
Ratio (E-t)/(hbar/2)                                         1.000000

------------------------------------------------------------------------------
--
```

```
sigma_r (coherent state)                                    2.194469e+22
m
sigma_r / R                                                     0.559414

delta_r_min = hbar*c/E1                                     3.103448e+22
m
delta_r_min / R                                                 0.791131

--------------------------------------------------------------------------------
--
delta(r)                                                   r/(2R)  exact

delta(r=R)                                                      0.500000

F_mod(R) / F_Newton(R)                                          0.500000

F_mod(2R)  [N/kg]                                               0.000000
N/kg
--------------------------------------------------------------------------------
--

================================================================
 PLOT 14: Physical Constants Hierarchy (log10 scale)
================================================================
```

PLOT 14 — Physical Constants Hierarchy: log10(value in SI)
Key quantities of the graviton-Newton overlap framework [Part IX]



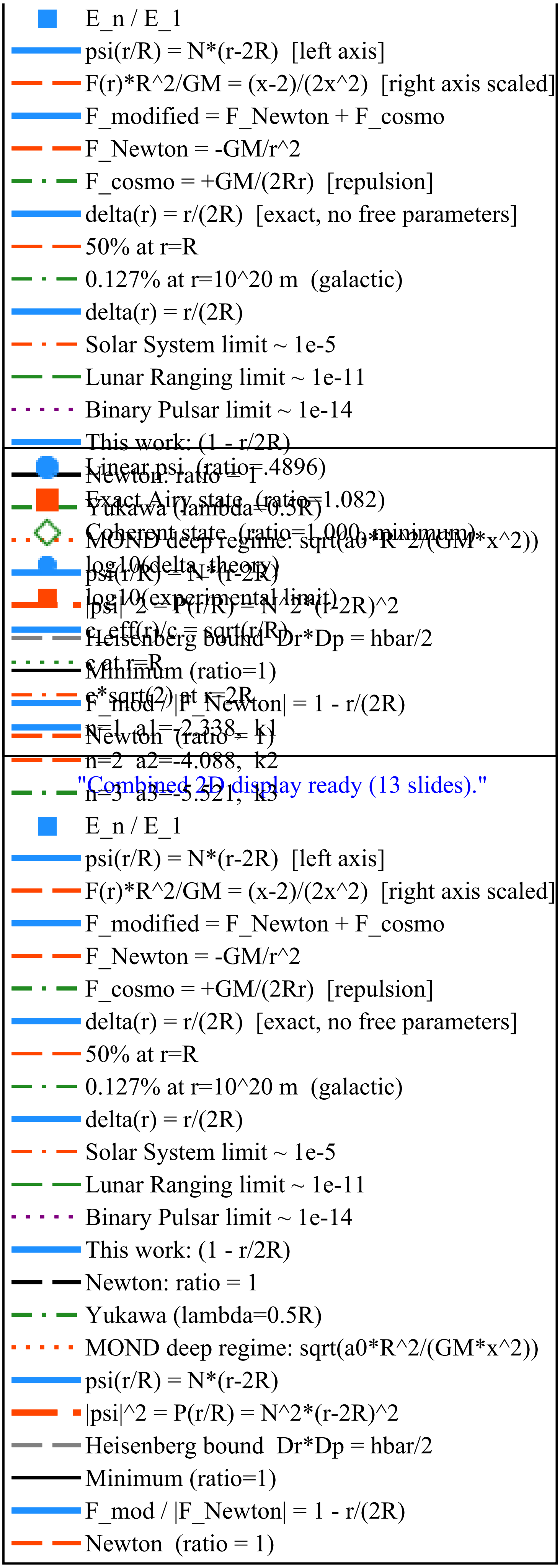
"PLOT 14 done."



==

COMBINED DISPLAY (2D — insequence)
Use arrow keys in Maple to cycle through all 13 two-dimensional plots.

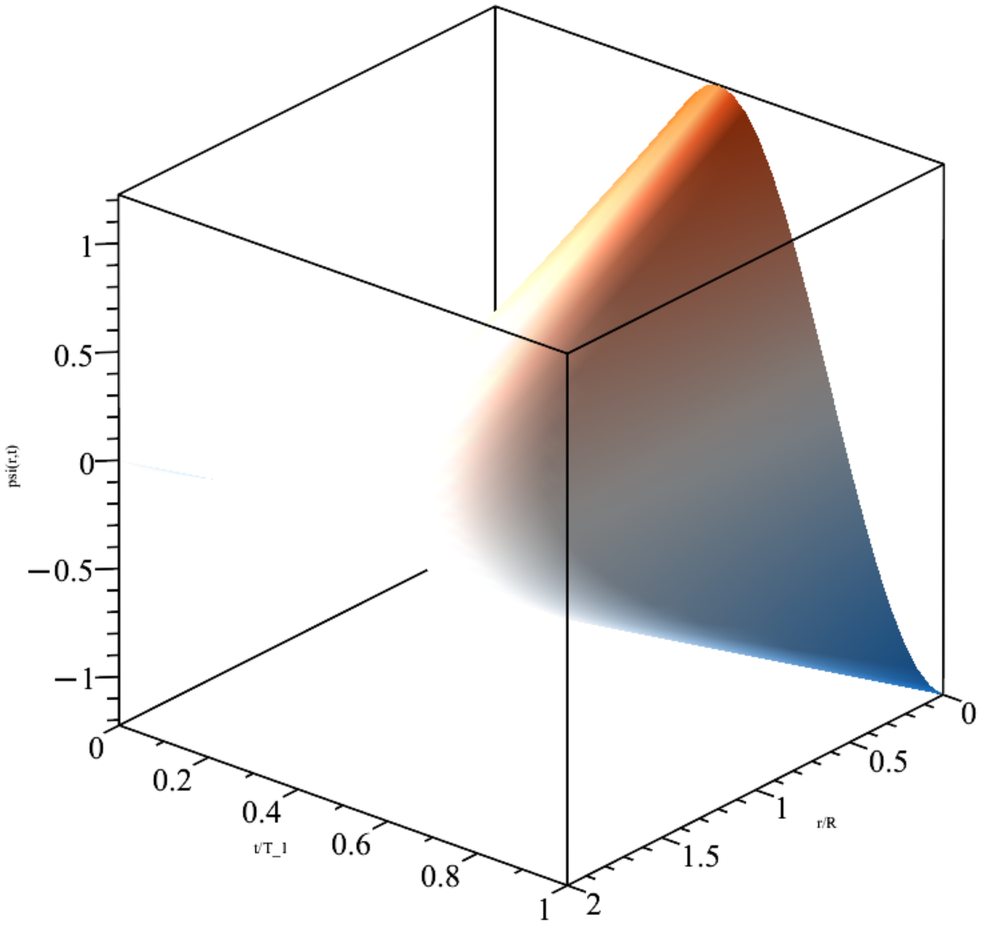
==


E_n / E_1
psi(r/R) = N*(r-2R) [left axis]
F(r)*R^2/GM = (x-2)/(2x^2) [right axis scaled]
F_modified = F_Newton + F_cosmo
F_Newton = -GM/r^2
F_cosmo = +GM/(2Rr) [repulsion]
delta(r) = r/(2R) [exact, no free parameters]
50% at r=R
0.127% at r=10^20 m (galactic)
delta(r) = r/(2R)
Solar System limit ~ 1e-5
Lunar Ranging limit ~ 1e-11
Binary Pulsar limit ~ 1e-14
This work: (1 - r/2R)
Linear psi (ratio=.4896)
Newton: ratio = 1
Exact Airy state (ratio=1.082)
Yukawa (lambda=0.5R)
Coherent state (ratio=1.000, minimum)
MOND deep regime: sqrt(a0*R^2/(GM*x^2))
log10(delta_theory)
psi(r/R) = N*(r-2R)
log10(experimental limit)
|psi|^2 = P(r/R) = N^2*(r-2R)^2
c_eff(r)/c = sqrt(r/R)
Heisenberg bound Dr*Dp = hbar/2
c at r=R
Minimum (ratio=1)
c*sqrt(2) at r=2R
F_mod / |F_Newton| = 1 - r/(2R)
n=1 a1=-2.338, k1
Newton (ratio = 1)
n=2 a2=-4.088, k2
"Combined 2D display ready (13 slides)."
n=3 a3=-5.521, k3
E_n / E_1
psi(r/R) = N*(r-2R) [left axis]
F(r)*R^2/GM = (x-2)/(2x^2) [right axis scaled]
F_modified = F_Newton + F_cosmo
F_Newton = -GM/r^2
F_cosmo = +GM/(2Rr) [repulsion]
delta(r) = r/(2R) [exact, no free parameters]
50% at r=R
0.127% at r=10^20 m (galactic)
delta(r) = r/(2R)
Solar System limit ~ 1e-5
Lunar Ranging limit ~ 1e-11
Binary Pulsar limit ~ 1e-14
This work: (1 - r/2R)
Newton: ratio = 1
Yukawa (lambda=0.5R)
MOND deep regime: sqrt(a0*R^2/(GM*x^2))
psi(r/R) = N*(r-2R)
|psi|^2 = P(r/R) = N^2*(r-2R)^2
Heisenberg bound Dr*Dp = hbar/2
Minimum (ratio=1)
F_mod / |F_Newton| = 1 - r/(2R)
Newton (ratio = 1)

```
===============================================================
 PLOT 10 (3D) — Graviton Field psi(r,t) — displayed separately
===============================================================
```

"3D spacetime surface:"

PLOT 10 — Graviton Field psi(r,t) = N*(r-2R)*cos(omega1*t) [3D]
x=r/R in [0,2], tau=t/T_1 in [0,1]   [Part V, 5.3]




```
===============================================================
 TRANSCRIPT COMPLETE
 Plots: PLOT01..PLOT14  (13 two-dimensional + 1 three-dimensional)
 Source: graviton_newtonian_overlap_EN.md
===============================================================
```